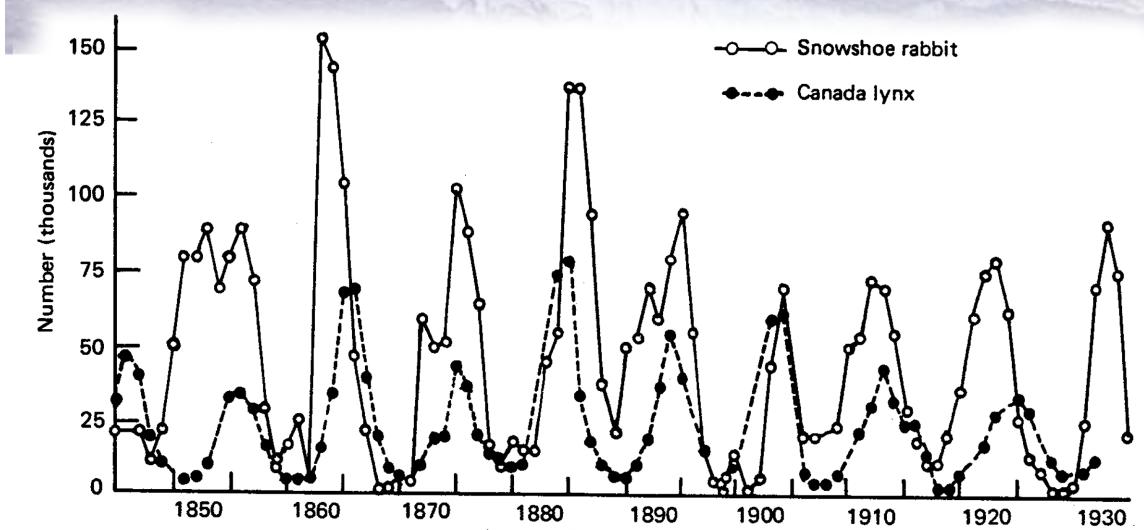


Projektarbeit: Das Räuber-Beute-Modell

pray-predator-model



Tobias Möhle
211204256

28.01.2013

Inhaltsverzeichnis

Einleitung	1
Deterministische Betrachtung	2
Betrachtung mit der Master-Gleichung	3
Berechnungen mit dem Gillespie-Algorithmus	5
Auswertung	10
Abbildungsverzeichnis	10

Einleitung

Das Räuber-Beute-Modell ist ein Modell aus der Ökologie, welches das kleinste ökologische System beschreibt: zwei Populationen; eine Räuber- und eine Beutepopulation. Wie sich diese Populationen gegenseitig beeinflussen, bzw. wie sich dieser Einfluss beschreiben lässt, soll im Folgenden betrachtet werden. Eine Betrachtungsweise ist ein deterministisches Modell, welches durch die Lotka-Volterra-Gleichungen¹

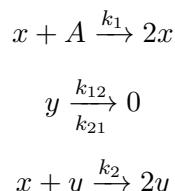
$$\begin{aligned}\dot{x} &= x(k_1A - k_{12}y) = 0 \\ \dot{y} &= y(k_{21}x - k_2) = 0\end{aligned}$$

beschrieben wird. Hierbei soll x die Beute- und y die Räuberpopulation darstellen; die Parameter haben die Bedeutung

- k_1A ist eine Größe, welche die Fertilität der Beutetiere beschreibt. Diese wird einerseits durch populationsspezifische Größen (Häufigkeit der Geburten, Anzahl der Kinder je Geburt etc.), welche in k_1 zusammengefasst sind, und andererseits durch die vorhandene Nahrungsmenge, symbolisiert mit A , beschrieben.
- k_2 beschreibt die Mortalität der Räuber
- k_{21} und k_{12} sind Größen, welche die Wechselwirkung der Populationen beschreibt, also die Anzahl der gerissenen Beutetiere auf der einen Seite und die Fertilität der Räuber auf der anderen Seite (welche von deren Nahrungsgrundlage, den Beutetieren, abhängt).

Die Lotka-Volterra-Gleichungen sind nicht allgemein lösbar, jedoch kann man an ihnen Betrachtungen zum Langzeit- und Stabilitätsverhalten durchführen.

Eine analoge Beschreibung wird durch die drei Reaktionsgleichungen



gezeigt. Ihr liegt, wie allen chemischen Prozessen, ein stochastisches Modell zu Grunde, welches mit Hilfe entsprechender Methoden analysiert werden kann. Eine Betrachtung durch die Master-Gleichung und eine Analyse mit Hilfe des Gillespie-Algorithmus' werden im folgenden vorgestellt.

¹<http://de.wikipedia.org/wiki/Lotka-Volterra-Gleichungen> (am 25.01.2013)

Deterministische Betrachtung

Die deterministische Betrachtung des Räuber-Beute-Systems folgt, wie bereits oben erwähnt, den beiden gekoppelten Differenzialgleichungen 1. Ordnung

$$\dot{x} = x(k_1 A - k_{12}y) = 0$$

$$\dot{y} = y(k_{21}x - k_2) = 0$$

Da diese analytisch nicht lösbar sind, werden hier lediglich die stationären Lösungen, das heißt Lösungen, bei welchen sich das System nicht mehr verändert und entsprechend Langzeitlösungen sind, betrachtet.

Ein stationärer Fall ist eingetreten, wenn $\dot{x} = \dot{y} = 0$. Bei den Lotka-Volterra-Gleichungen ergeben sich zwei stabile Punkte: $x_1 = 0, y_1 = 0, x_2 = \frac{k_2}{k_{21}}, y_2 = \frac{k_1 A}{k_{12}}$.

Nun gilt es, zu untersuchen, wie stabil das System gegenüber kleinen Schwankungen an diesen Punkten ist. Dies lässt sich mit der Gleichung $\delta \begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} = J\delta \begin{pmatrix} x \\ y \end{pmatrix}$ wobei J die Jakobimatrix

$$J = \begin{pmatrix} k_1 A - k_{12}y & -k_{12}x \\ k_{21}y & k_{21}x - k_2 \end{pmatrix}.$$

ist, untersuchen.

Für x_1, y_1 gilt also: $\begin{vmatrix} -\lambda & -k_{12}\frac{k_2}{k_{21}} \\ k_{21}\frac{k_1 A}{k_{12}} & -\lambda \end{vmatrix} = \lambda^2 + k_1 k_2 A$.

Dieses System hat lediglich die imaginären Lösungen $\lambda = \pm i\sqrt{k_2 k_1 A}$. Rein imaginäre Lösungen beschreiben weder stabile noch instabile Systeme sondern ozillierende Lösungen (analog zu Schwingungsgleichungen in der Mechanik).

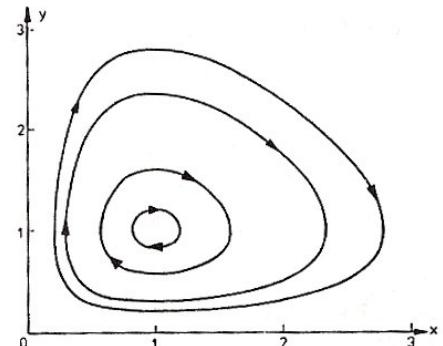
Dabei verschwindet diese Oszillation, wenn gilt:

Andere Lösungen oszillieren entsprechend um diesen stabilen Punkt.

Es lässt sich weiter zeigen, dass, analog dem mathematischen Pendel oder anderen oszillierenden Systemen, hier eine Erhaltungsgröße analog der Energie bzw. Hamilton-Funktion eingeführt werden kann^a. Diese lautet:

$$H = k_{21}x - k_2 \ln(x) + k_{12} - k_1 A \ln(y) = \text{const}$$

Im Phasenraum ergeben sich damit nebenstehend dargestellte Trajektorien. Dabei entsprechen die unterschiedlichen Ringe unterschiedlichen Werten für H .



^aR. Mahnke „Nichtlineare Physik in Aufgaben“ Verlag: B.G. Teubner, Stuttgart 1994; S. 125

Abbildung 1: Dynamik des Räuber-Beute-Systems im Phasenraum

Betrachtung mit der Master-Gleichung

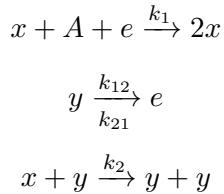
Es gibt zwei unterschiedliche Master-Gleichungen, welche prinzipiell das gegebene System beschreiben: zum einen ist das die diskrete Master-Gleichung

$$\frac{\partial}{\partial t} P(n, m, t) = k_1 A(n-1)P(n-1, m, t) + k_{21}(n+1)(m-1)P(n+1, m-1, t) + k_2(m+1)P(n, m+1, t) -$$

$$[k_1 An + k_{21}nm + k_2m] P(n, m, t)$$

bei welcher n die Größe der Beute-Population und m die der Räuber-Population ist. Diese ist jedoch nicht analytisch lösbar, da es hier zunächst unendlich viele miteinander gekoppelte Gleichungen gibt. Dieses Problem ist zwar durch die Einführung einer maximalen Populationsanzahl durchaus zu beheben, jedoch wird das System nicht handlicher, da bereits bei einer Maximalpopulation von 3 Tieren 10 gekoppelte Differentialgleichungen zu lösen sind.

Diese Gleichung lässt sich nun jedoch in eine kontinuierliche Gleichung überführen, indem man von einer maximalen Gesamtpopulation V ausgeht, welche nicht überschritten werden kann. Damit ändern sich die grundlegenden Reaktionsgleichungen zu²



dabei ist e die Anzahl "freier" Plätze in dem System. Damit muss sich auch die erste Übergangsgröße verändern. Aus $k_1 An$ wird nun $k_1 Ane$. Für den Kontinuitätsübergang werden nun noch die Größen $x = \frac{n}{V}$, $y = \frac{m}{V}$ eingeführt. Entsprechend gilt nun für die Wahrscheinlichkeiten:

$P(n, m, t) = P(xV, yV, t) \rightarrow p(x, y, t) V$. Wir erhalten also eine Wahrscheinlichkeitsdichte. Auch die Übergangsraten www.co-nan.eu/pdf/df2.pdf müssen nun angepasst werden. Dabei formen wir um:

- $k_1 A(n-1)(e+1) = k_1 A V^2 \frac{n-1}{V} \frac{V-(n-1)-m}{V}$ wegen des Zusammenhangs $V = m+n+e$. Nun folgt nach Definition mit $w_1 = k_1 A V^2$: $k_1 A(n-1)(e+1) = w_1 x'(1-x'-y)$. Analog folgen die weiteren Größen:
 - $k_2(m+1) = w_2 y'$ mit $w_2 = k_2 * V$
 - $k_{21}(m-1)(n+1) = w_3 x' y'$ mit $w_3 = k_{21} V^2 = w_3$.

Damit folgt die Master-Gleichung:

$$\begin{aligned} \frac{\partial}{\partial t} p(x, y, t) &= w_1 x'(1-x'-y)p(x', y, t) + w_2 y' p(x, y', t) + w_3 x' y' p(x', y', t) \\ &\quad - (w_1 x(1-x-y) + w_2 y + w_3 xy) p(x, y, t) \end{aligned}$$

Diese Gleichung soll nun auf die Form

$$\frac{\partial}{\partial t} p(x, t) = \int dx' w(x, x') p(x', t) - \int dx' w(x', x) p(x, t)$$

in der die Master-Gleichung bekannt ist, gebracht werden. Durch Koeffizientenvergleich erhält man für die nun zweidimensionalen Übergangsraten:

$$w(x, y, x', y') = w_1 x'(1-y'-x')\delta(x' - \frac{n-1}{V})\delta(y' - y) + w_2 y'\delta(x' - x)\delta(y' - \frac{m+1}{V}) + w_3 x' y' \delta(x' - \frac{n+1}{V})\delta(y' - \frac{m-1}{V})$$

² vergleiche auch <http://www.co-nan.eu/pdf/df2.pdf> (am 21.12.2012)

$$w(x', y', x, y) = \{w_1 x(1 - x - y) + w_2 y + w_3 x y\} \delta(x' - x) \delta(y' - y)$$

Damit erhalten wir schließlich die kontinuierliche Master-Gleichung:

$$\begin{aligned} \frac{\partial}{\partial t} p(x, y, t) &= \int dx' w(x, y, x', y') = w_1 x'(1 - y' - x') \delta(x' - \frac{n-1}{V}) \delta(y' - y) \\ &\quad + w_2 y' \delta(x' - x) \delta(y' - \frac{m+1}{V}) + w_3 x' y' \delta(x' - \frac{n+1}{V}) \delta(y' - \frac{m-1}{V}) \\ &- \int dx' w(x', y', x, y) = \{w_1 x(1 - x - y) + w_2 y + w_3 x y\} \delta(x' - x) \delta(y' - y) \end{aligned}$$

Die Diskrete Master-Gleichung hat gegenüber der kontinuierlichen jedoch zwei Vorteile: im Wesentlichen ist sie deutlich anschaulicher, da statt mit Dichten und Anteilen mit „realen“ Größen gerechnet wird. Aber auch ihre Handhabung ist deutlich leichter, da dort keine Distributionen oder ähnliches vorkommt. Hinzu kommt, dass die Master-Gleichung auch im kontinuierlichen Fall nicht lösbar ist, da jede Gleichung drei Wahrscheinlichkeiten miteinander koppelt. Dieses Problem soll im Folgenden durch einen Algorithmus umgangen werden, welcher die Master-Gleichung numerisch löst.

Berechnungen mit dem Gillespie-Algorithmus

Der Gillespie-Algorithmus untersucht stochastische Prozesse mit einem statistischen Ansatz: Es werden eine Anzahl von Durchläufen des Prozesses mit Hilfe von Zufallsfunktionen berechnet und anschließend über diese gemittelt. Wurden genügend Durchläufe gemacht und sind die verwendeten Zufallsalgorithmen gut, so entsprechen die mit dem Gillespie-Algorithmus gewonnenen Ergebnisse denen der deterministischen Lösung. Das genaue Vorgehen beim Gillespie-Algorithmus lautet wie folgt³:

1. Initialisiere zunächst alle Startwerte (Anfangsbedingungen, Konstanten etc). In meiner Implementierung wird dieser Schritt separat bereits im constructor des Programms vorgenommen.
2. Generiere zwei Zufallszahlen τ , ρ , wobei τ eine exponentiell verteilte Zahl sein soll: Kleine Zahlen sollen also sehr häufig vorkommen, größere immer seltener (wird zum Beispiel, wie in meiner Implementierung, durch eine gleich-verteilte Zufallsvariable erzeugt, welche einer e-Funktion übergeben wird.). ρ soll eine Zahl zwischen 0 und 1 sein. Die Werte von ρ sind gleichverteilt (entsprechende Algorithmen bietet jede höhere Programmiersprache). In meiner Implementierung kommen keine Zeitsprünge größer als 1 Zeiteinheit vor:

```
double changetime=exp(-double(rand())/double(rand()))
double reaction=double(rand())/RAND_MAX;
```

3. Nun wird ein Reaktionsschritt durchgeführt: Die Zeit wird um die Variable τ weiter gesetzt (Zeitpunkt, zu dem die Reaktion stattfindet) und ρ wird auf die möglichen Reaktionsschritte abgebildet.

Da bei dem Räuber-Beute-Modell die Wahrscheinlichkeit von den Populationsgrößen abhängt, wird hier jeweils die Reaktionskonstante (k_1, k_2, k_{21}) mit der entsprechenden Populationsgröße addiert und durch die Gesamtgröße geteilt. In meiner Implementierung:

```
double foo=k1*prepre[i].prey[j];
double bar=foo+k2*prepre[i].pred[j];
double foobar=bar+(k21+k12)*prepre[i].pred[j]*prepre[i].prey[j]/2;
```

Nun kann durch Vergleich der Größen einer der Prozesse durchgeführt werden:

```
if (reaction<=foo/foobar){ //a new prey is born
else if(reaction<=bar/foobar){ // a pred died
else { //a pred ate a prey
```

4. Ist die zu Beginn festgelegte Simulationsdauer noch nicht erreicht, beginne wieder bei Schritt 2. Dabei muss darauf geachtet werden, dass die Wahrscheinlichkeiten beim nächsten Reaktionsschritt anders verteilt sind.

Die gesamte Implementierung des Algorithmus' kann auf der Seite

<https://github.com/BalticPinguin/Prey-Predator-System.git>

eingesehen und heruntergeladen werden.

Mit Hilfe dieses Algorithmus' lässt sich das Räuber-Beute-Modell nun gut beschreiben. Allerdings stellt sich die Suche geeigneter Parameter als deutlich schwerer dar, als man es zunächst erwarten würde, da die Zusammenhänge, wie man bereits anhand der Graphiken erahnen kann, nicht linear sind. Stirbt beispielsweise die Räuber-Population bei den Simulationen stets aus, hilft es nicht, ihr Sterberate einfach zu senken; vielmehr hilft ihnen eine höhere Geburtenzahl

³<http://www.co-nan.eu/pdf/df2.pdf> (28.01.2013)

bei den Beutetieren, da sie sich so besser erholen und damit die Räuber-Population stärken.

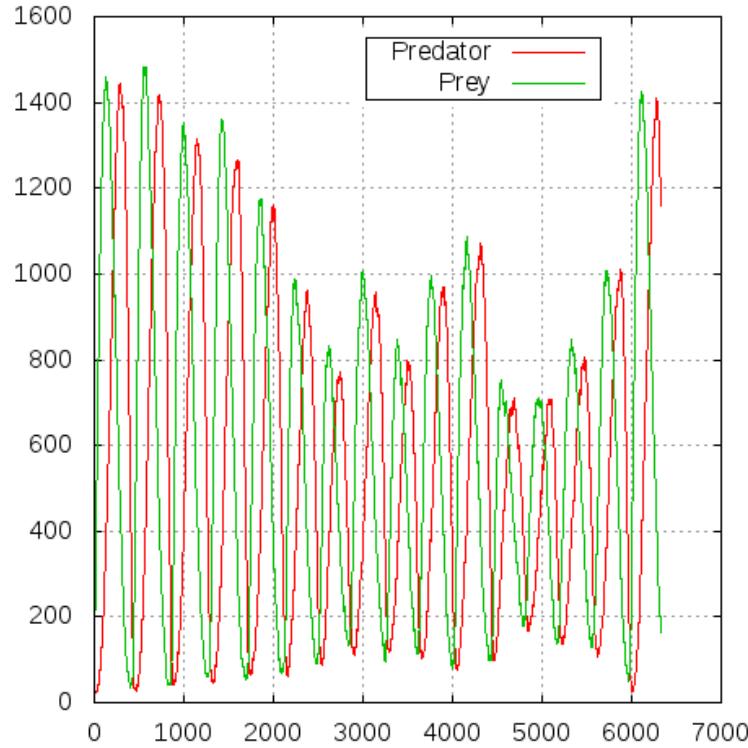


Abbildung 2: zeitliche Entwicklung der Populationen (Einzeltrajektorie)

Die nebenstehenden Graphiken zeigen eine mit dem Algorithmus berechnete Einzeltrajektorie; einmal als zeitlicher Verlauf (oben) und einmal im „Phasenraum“, welcher von den Populationgrößen aufgespannt wird. In diesem sieht man also die Kopplung der beiden Größen untereinander. Auch zeigt sich hier, abgesehen von dem spiralförmigen Verlauf, dass der berechnete Prozess offensichtlich dem oben betrachteten deterministischen Modell ähnlich ist, da die Prozesse in der gleichen leicht dreieckigen Form ablaufen.

Allerdings ist die hier abgebildete Trajektorie für die in diesem Prozess genutzten Parameter eher untypisch. Mittelt man hier über viele Prozesse, so zeigt sich, dass meist die Räuber-Population mehr oder weniger bald ausstirbt. Dadurch, dass es sich hier um eine Einzeltrajektorie handelt, sieht man hier aber sehr gut die stochastischen Schwankungen, denen der Zufallsprozess unterliegt.

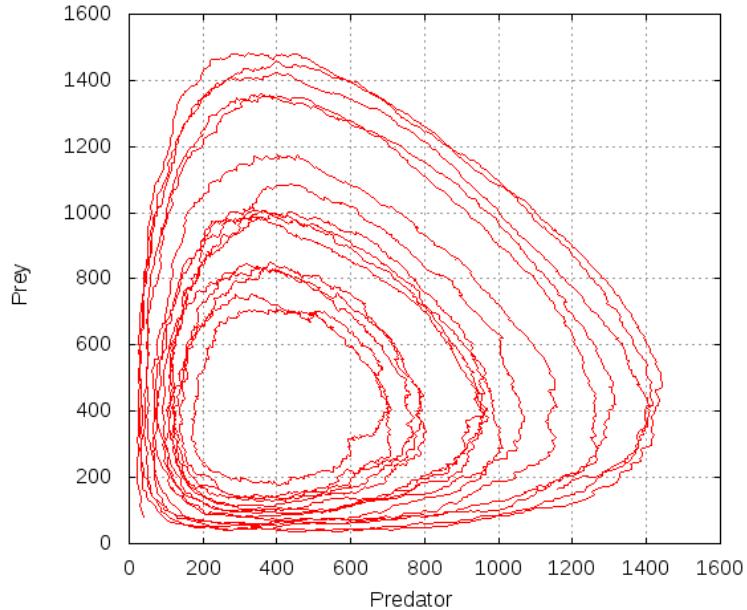


Abbildung 3: Entwicklung der beiden Populationen im Phasenraum (Einzeltrajektorie)

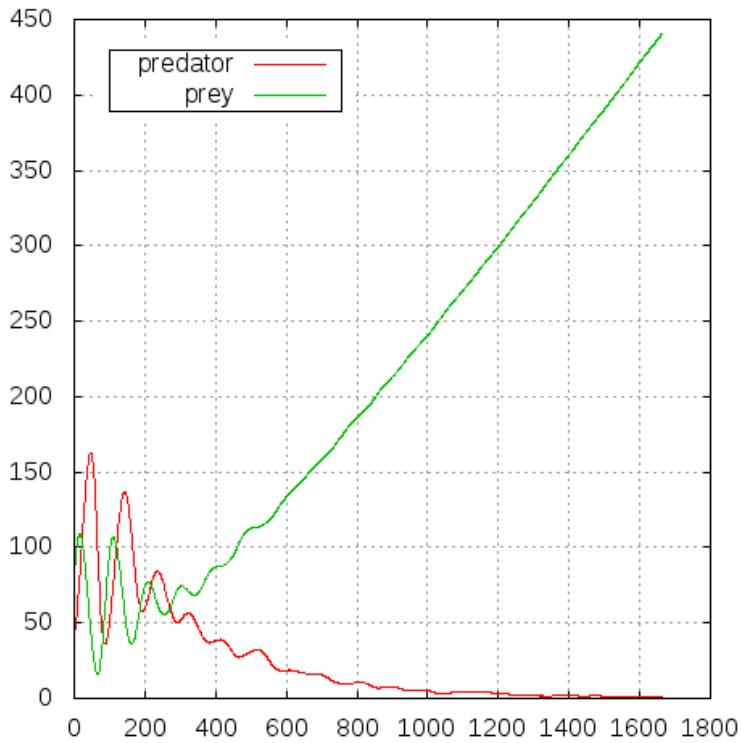


Abbildung 4: zeitliche Populationsentwicklung, Räuber-Population ausstirbt

Die hier abgebildeten Graphiken (oben zeitlicher Verlauf, unten Trajektorie im Phasenraum) entstanden aus einer Mittlung über 1000 Simulationsdurchläufe. Dabei wurden die Parameter hier so gewählt, dass sie die in der deterministischen Betrachtung oben gefundenen Bedingungen $x = \frac{k_2}{k_{21}}$, $y = \frac{k_1 A}{k_{12}}$ genügen. Offensichtlich unterscheidet sich dieses Modell von dem deterministischen quantitativ recht stark. Die Parameter haben offensichtlich in diesem Modell doch eine etwas andere Bedeutung als in dem deterministischen Modell. Allerdings scheinen sie in ihrer Proportionalität denen oben zu entsprechen. So sind auch bei stabileren Durchläufen die Konstanten k_{21} und k_{12} wenn um einige Größenordnungen kleiner als die anderen beiden.

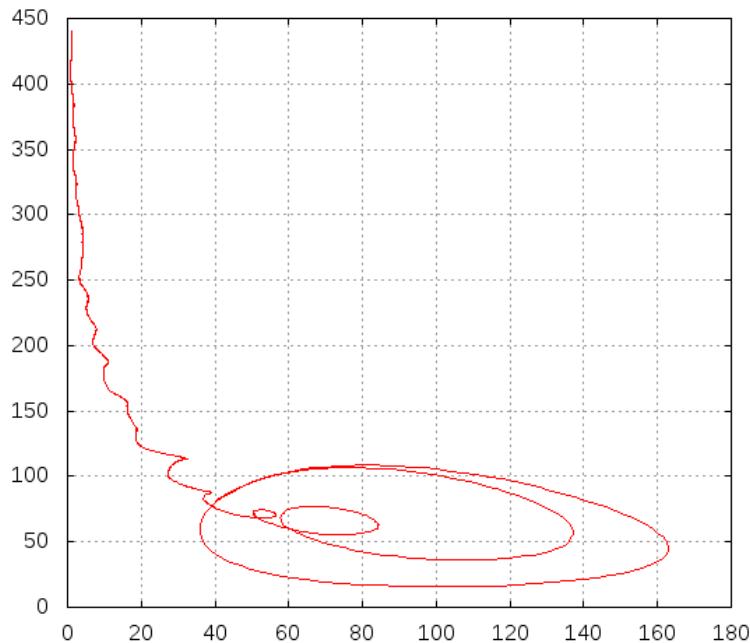


Abbildung 5: obige Entwicklung im Phasenraum

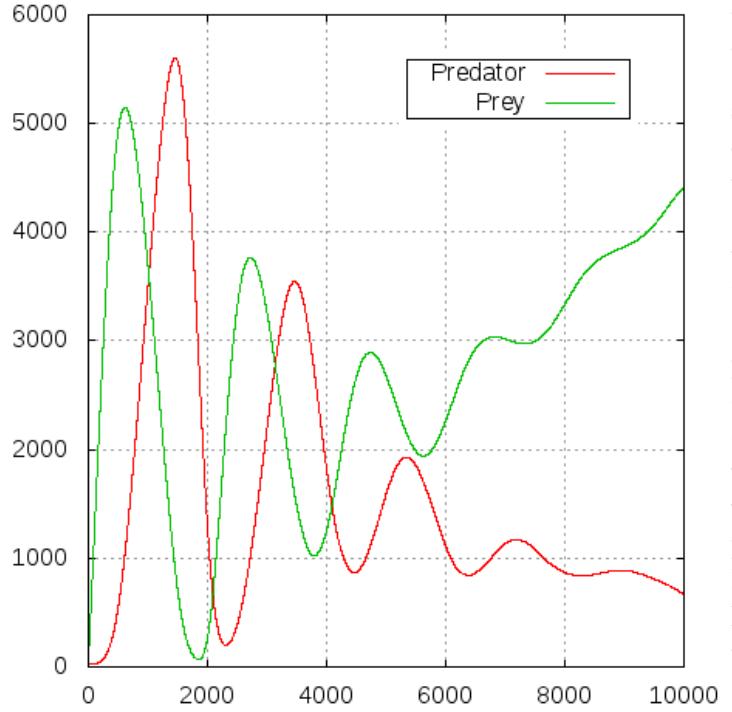


Abbildung 6: gemittelte Trajektorie in einem fast stabilen unterschiedlich schnell oszillierten und Zustand

Die nebenstehenden Bilder zeigen ein allgemeines Problem dieser Auswertungsmethode. Im Wesentlichen sind die Parameter offensichtlich so gewählt, dass die Räuber-Population gegen Ende häuft ausstirbt, da die Beute entsprechend ansteigt. Viel wesentlicher ist aber, dass, wie auch beim ersten Beispiel zu sehen ist, das System hier offensichtlich gedämpft ist. Da die Dämpfung bei vielfach gemittelten Ergebnissen deutlich stärker ist und, wie das unterste Bild zeigt, die Standardabweichung mit der Zeit exponentiell zunimmt, können hier zu der ohnehin anscheinend stattfindenden Dämpfung zwei Gründe vermutet werden: Erstens kann es sein, dass bei vielen der Simulationen eine der Populationen ausgestorben ist; die mittlere Populationszahl jedoch annähernd konstant ist (was die sehr glatte Funktion der Standardabweichung von der Zeit erklärt). Ein zweiter Grund könnte sein, dass die einzelnen Simulationen

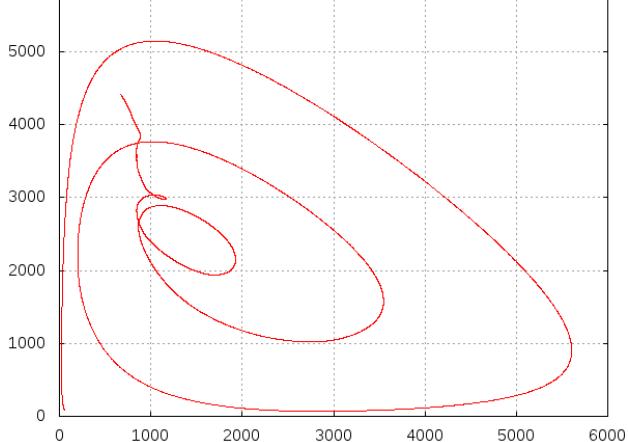


Abbildung 7: Trajektorie im Phasenraum

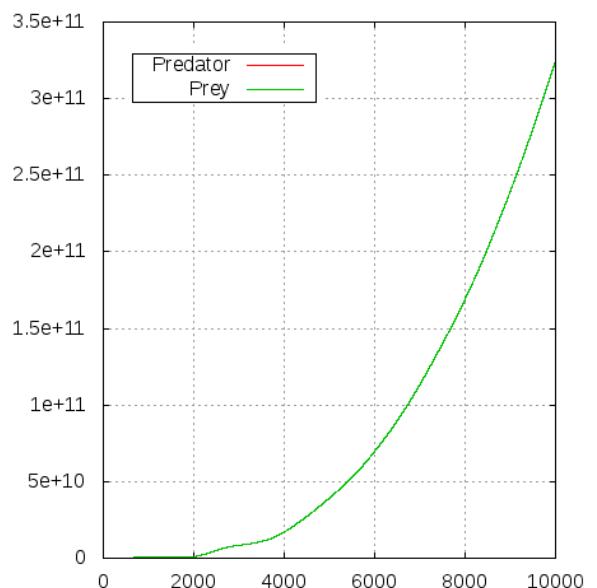


Abbildung 8: zeitliche Entwicklung der Standardabweichung im fast stabilen Zustand

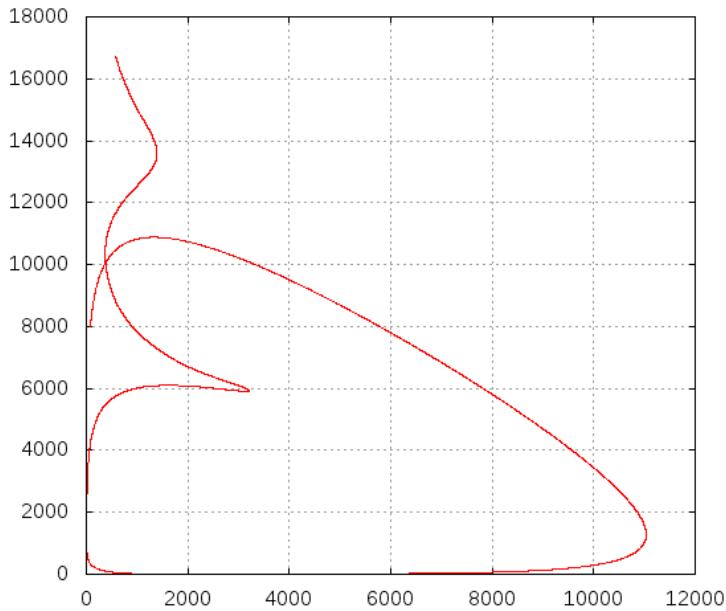


Abbildung 9: Populationsentwicklung bei anderer Parameterwahl

Ein sehr interessanter Effekt zeigt sich, wenn man ganz andere Parameter eingibt als die oben genutzten. Offensichtlich enthält die Gillespie-Simulationen Bereiche, die ebenfalls recht stabil ablaufen, jedoch anders verlaufen als die oben betrachteten und aus anderen Modellen bekannten Modelle.

Die Untersuchung solcher Systeme und Klärung der Frage, woher solch ein unregelmäßiges Verhalten kommt, sowie ob dieses Verhalten die Grenze des Modells markiert oder auch in anderen Modellen zu finden oder gar mit der Realität übereinstimmen kann, würde den Rahmen dieser Arbeit sprengen.

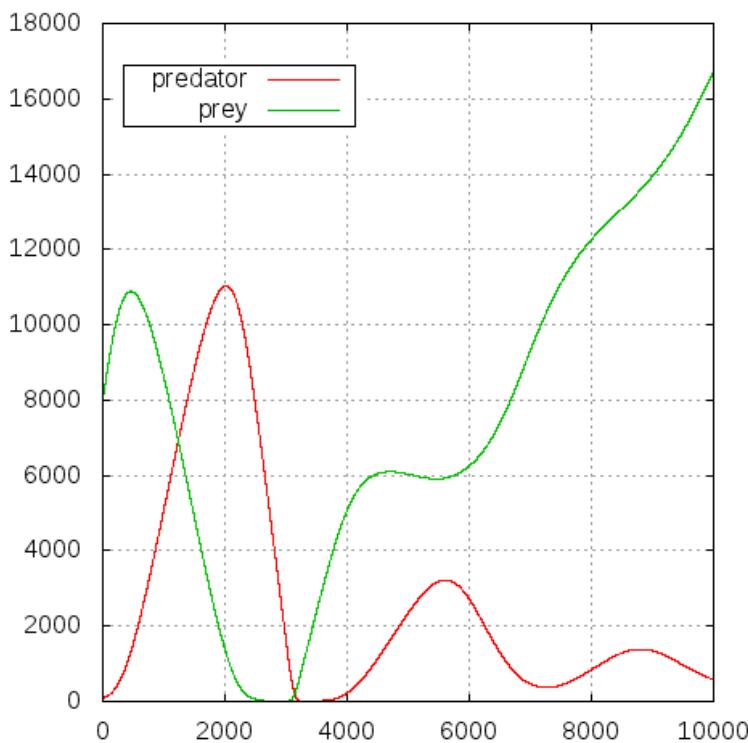


Abbildung 10: die obige Populationsentwicklung im Phasenraum: Das besondere Verhalten wird deutlicher

Auswertung

Das Räuber-Beute-Modell ist offensichtlich ein sehr ergibiges Problem, welches sich, wie die obigen Betrachtungen zeigen, auf verschiedene Weise beschreiben und untersuchen lässt. Dabei hat jede Untersuchungsweise ihre Vor- und Nachteile; aber es konnte auch gezeigt werden, dass die beiden hier untersuchten Modelle ähnliche Ergebnisse liefern.

Abbildungsverzeichnis

- Quellen Titelbild: http://iqa.evergreenps.org/science/biology/predator-prey_files/pred-prey.jpg (13.02.2013), <http://www.globalchange.umich.edu/globalchange1/current/lectures/predation/tmp26.gif> (04.01.2013)
- Abbildung 1: R. Mahnke „Nichtlineare Physik in Aufgaben“ Verlag: B.G. Teubner, Stuttgart 1994; S. 125
- Abbildung 2-3: Parameter: 80, 40, 6500, 4.8, 2.0, 0.01, 0.01, 1
- Abbildung 4-5: Parameter: 80, 40, 1700, 5.6, 2.8, 0.07, 0.07, 1000
- Abbildung 6-8: Parameter: 80, 60, 50000, 11.2, 8.1, 0.011, 0.011, 1000
- Abbildung 9-10: Parameter: 8000, 80, 50000, 17.3, 16.4, 0.013, 0.013, 1000

*Hinweis zu der Parameterangabe: Reihenfolge der angegebenen Zahlen:
Beutepopulation, Räuber-Population (jew. Startwerte), zu durchlaufende Zeiteinheiten, k_1 , k_2 ,
 k_{21} , k_{12} , Anzahl der berechneten Simulationen*