

Simulation eines Bienenstocks



Einleitung

1) Definition des physikalischen Bereichs:

Wir betrachten ein beschränktes Feld mit einem Bienenstock, der Bienen enthält. Ziel des Projekts ist es, den optimalen Flugradius zu ermitteln, den eine Biene im Durchschnitt fliegt, sodass der Nektarertrag des gesamten Stocks maximiert wird.

2) Symbolische Beschreibung des Modells:

Die Simulation wird mit C++ durchgeführt; das Feld, der Bienenstock und die Bienen werden als Klassen implementiert.

Das Feld ist in der Simulation die „Basis-Klasse“, auf der die Simulation durchgeführt wird – es besteht aus diskreten Punkten (Blumen). Im Bienenstock sind prinzipiell nur Nektar und Bienen enthalten; die Bienen lassen sich als wandernde Punkte auf dem Feld interpretieren.

3) Untersuchung des Modells:

Der Hauptanteil des Projekts ist also eine Computersimulation (Bottom-Up-Modell), welche für jede einzelne Biene ihre aktuelle, individuelle Aufgabe in einzelnen Schritten ausführt.

Unabhängig davon wird eine analytische Lösung eines passenden Top-Down-Modells nur kurz besprochen.

4) Vergleich mit bekannten Daten:

Das Ergebnis lässt sich mit bekannten Daten aus der Literatur vergleichen; wie sich zeigen wird, erhalten wir ein Resultat in einem sinnvollen Bereich.

Bottom-Up-Modellierung

Variante 1:

Feld:

Auf einem quadratischen Feld wird mittig ein Bienenstock platziert. Das Feld hat eine Auflösung von 1 [m²].

Mithilfe eines Zufallsgenerators und einer übergebenen Wahrscheinlichkeit werden auf dem Feld zufällig Blumen verteilt. Jede Blume hat zu Beginn einen "Startnektar".

Der Nektarnachwuchs der Blumen wird durch eine Funktion ausgeführt die in "regelmäßigem" Abstand aufgerufen wird.

Bienenstock:

Der Stock hat die Koordinaten des Feldmittelpunkts.

Im den Stock wird eine fixe Anzahl an Bienen gesetzt.

Weiters hat er die Menge des bereits gesammelten Nektars enthalten.

Biene:

Die Biene enthält nur den aktuellen Ort.

Simulation:

Jede Biene ist zu Beginn der Simulation im den Bienenstock.

Es wird eine Schleife durchlaufen, in der für jeden Schritt jede Biene folgendes ausführt:

Die Biene führt einen 2D-Zufallsflug aus.

D.h.: Wenn das Feld als 2D-Vektorraum aufgefasst wird, addiert der Computer per Zufallsgenerator einen der folgenden Vektoren: (1,0), (0,1), (0,-1) oder (-1,0).

Dann wird überprüft, ob bei den Koordinaten, an denen sich die Biene befindet, auch eine Blume mit Nektar ist.

Sollte dies zutreffen, so wird im Bienenstock eine Nektareinheit hinzugezählt und auf der Blume eine entfernt. Weiters wird die Biene in den Bienenstock zurückgesetzt.

Wenn die Bedingung nicht zutrifft, wird nichts verändert!

Diskussion der Variante:

Grundsätzlich wurde die Variante erstellt, um festzustellen, ob wir das Konzept in die Tat umsetzen können. Der offensichtlichste Mangel ist natürlich die Tatsache, dass alle Bienen immer im Bienenstock starten und von dort per Zufall genau ein Feld in eine Richtung weiterfliegen. Außerdem fehlt noch die Auswertung der Daten, welche lediglich mit "cout" ausgegeben werden.

Variante 2:

Allgemein wird der Syntax "verschönert". Dies ist notwendig, da Variante 1 nur zum Testen des Konzepts gedacht war.

Feld:

Syntaxverbesserung beim Erstellen des Feldes. (nutze Pointer: `int **Blume`)

Bienenstock:

Keine bemerkenswerten Veränderungen vorgenommen.

Biene:

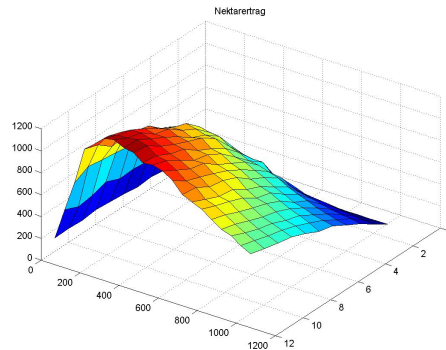
Die Bienen fliegen bei einem Start vom Bienenstock aus zuerst einen Grundradius in eine zufällige Richtung; von dort aus beginnen sie die 2D-Zufallssuche wie in Variante 1. Dieser Grundradius wird zu Beginn der Simulation übergeben und dann nicht mehr verändert.

Simulation:

Vom Konzept gleich Variante 1, jedoch wird der gefundene Nektar entsprechend einem Faktor (abhängig von der geflogenen Grunddistanz) reduziert.

Weiters werden die Daten in ein Textfile (rechteckige Anordnung) ausgegeben; dieses File kann dann in MATLAB eingelesen und entsprechend geplottet werden.

Ausgabe:



Diskussion der Variante:

In die Simulation ist in keiner Weise die Geschwindigkeit, die eine Biene fliegt, oder die Dauer eines Tages eingeflossen.

Außerdem wird nicht berücksichtigt, dass die Anzahl der Bienen im Laufe der Zeit steigt.

Variante 3:

Grundsätzlich ist es nun möglich, die Simulation für einen Zeitraum durchführen zu lassen, der in Tagen angegeben wird. Weiters fließt nun die Fluggeschwindigkeit der Biene in die Berechnung ein; diese kann vor der Simulation fixiert werden.

Der Syntax wurde erneut überarbeitet, um die typischen C++-Eigenschaften (private und public) zu nutzen; dies verringert auch mögliche Fehler im Programm.

Feld:

Grundsätzlich keine Veränderungen, jedoch könnte man nun ausgeben lassen, wie oft eine bestimmte Blume von Bienen besucht wurde. Dies ist jedoch bei der Auswertung durch MATLAB sehr zeitraubend!

Bienenstock:

Es kann nun zu Beginn der Simulation eine fixe "Wachstumszahl" übergeben werden. Die Anzahl der Bienen im Bienenstock wird zu Beginn eines jeden Tages um diese Zahl erhöht.

Biene:

Der Grundradius, den die Biene zu Beginn einer Suche fliegt, wird gespeichert. (In der Simulation fließt dieser nun besser ein, um den Verringerungsfaktor für den Honigertrag sowie die Dauer für den Flug besser in der Simulation berücksichtigen zu können.)

Simulation:

Die Simulation wird in Mehrfach-Schleifen durchgeführt. Für jeden zu simulierenden Grundradius wird für jeden Tag, jeden Schritt, den eine Biene pro Tag macht, und für jede Biene folgendes ausgeführt:

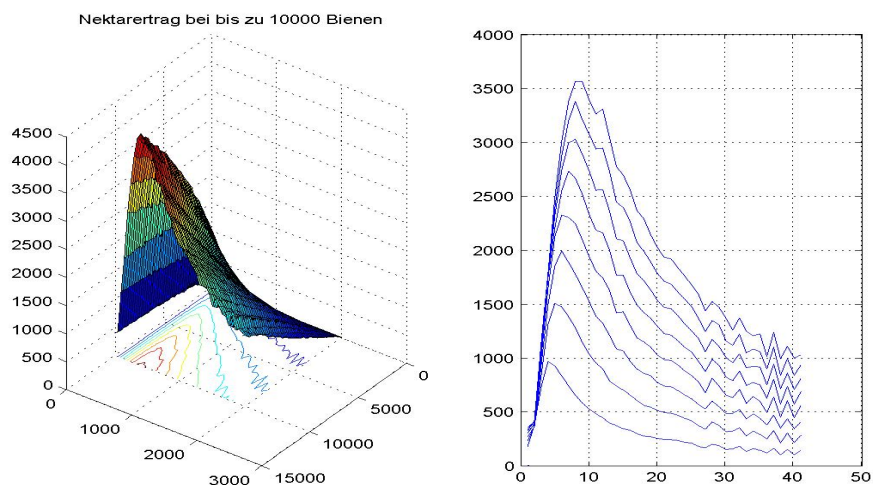
Beim ersten Schritt am Tag fliegt jede Biene einen Grundradius in eine beliebige Richtung. Die doppelte Distanz, die die Biene hierbei zurücklegt, wird an die Biene übergeben und die Variable "Pause" der Biene auf "true" gesetzt.

An späteren Tagen wird zuerst überprüft, ob die Biene eine Pause macht; wenn ja, dann wird die Distanz innerhalb der (Klasse) Biene um 1 verringert.

Dies wird in jedem Durchlauf der Schleife gemacht, solange bis die Distanz auf 0 ist. Dann wird die Variable Pause auf "false" gesetzt.

Wenn die Biene keine Pause macht, wird die Honigsuche analog zu Variante 2 durchgeführt.

Ausgabe:



Diskussion der Variante:

Das ändern des Startpunktes auf einen zufälligen Punkt eines vorgegebenen Startkreises ist nicht wirklich eine "passende" Simulation, da es sicher nicht zutrifft, dass alle Bienen im Stock immer denselben Radius fliegen.

Ein Beispiel: Die Bienen fliegen 1 [km], aber selbst wenn in unmittelbarer Nähe des Stocks eine Nektarquelle ist, würde diese von keiner Biene besucht werden.

Weiters ist in keiner Weise in die Simulation eingeflossen, dass nicht alle Blumen zur selben Zeit blühen bzw. dass nicht an allen Punkten auf einem Feld die gleiche Nektarmenge ist.

Variante 4:

Um die Rechengeschwindigkeit zu erhöhen sowie die potentiellen Feldgrößen zu vergrößern, wird ein Umrechnungsfaktor an die Simulation übergeben, welcher wie ein Maßstab einprogrammiert wurde. Hierbei ist es notwendig, den Startnektar auf Null zu setzen, da sonst durch die Skalierung dieser Zahl ein enorm hoher Startnektar zustande kommen würde, was vor allem bei kleinen Radien zu "Verzerrungen" der Daten führen kann.

Da C++ bei dem Versuch auf einen nicht deklarierten Speicher zuzugreifen abstürzt, wurde einprogrammiert, dass eine Biene, die den Rand erreichen würde, auf einen Platz weiter innen versetzt wird. Dieser Fehler ist jedoch vernachlässigbar, da das Feld ein Quadrat ist und die Ecken nur wenig genutzt werden; dieser Fehler tritt also nur in der Nähe der "Hauptachsen" auf!

Feld:

Es ist nun möglich einen Array von "Setzlingen" zu übergeben, die dann per Zufallsgenerator auf dem Feld verteilt werden. Jeder dieser Setzlinge hat nun verschiedene Daten für Wahrscheinlichkeit auf dem Feld, Nektarmenge, Beginn und Dauer der Blühperiode sowie die Wachstumsrate an Nektar pro Tag.

Das Wachstum für die Blumen ist nun natürlich auch davon abhängig, ob der jetzige Zeitpunkt innerhalb der Blühperiode liegt.

Bienenstock:

Keine bemerkenswerten Veränderungen vorgenommen.

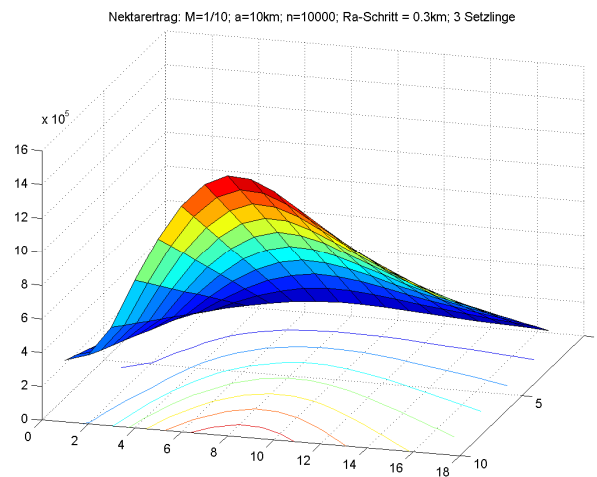
Biene:

Die am Beginn der Simulation übergebenen Suchradien sind nun die maximalen Radien die, eine Biene fliegt. Bei einer neuen Suche wird der Startort nun erstens in einer zufälligen Richtung und zweitens in einer zufälligen Entfernung kleiner der Maximaldistanz gewählt. Diese "Distanzwahl" wird so durchgeführt, dass die Bienen auf dem Feld theoretisch gleichverteilt sind.

Simulation:

Die Simulation läuft im Prinzip gleich wie in Variante 3 ab.

Ausgabe:



Diskussion der Variante:

Es ist bereits gelungen, die Simulation so zu programmieren, dass die wichtigsten Eigenschaften einer Biene einfließen. Jedoch ist es noch notwendig zu berücksichtigen, dass die Biene nur begrenzt individuell handelt, da ein Bienenstock beim Nektarsuchen als Kollektiv arbeitet. Diese biologische Fähigkeit der Bienen ist der "Bientanz". Das bedeutet, dass eine Biene, die eine gute Nektarquelle gefunden hat, anderen Bienen sagt, wo diese zu finden ist. Für unser Modell ist es nun interessant, diesen Informationsaustausch zu berücksichtigen.

Tanz der Bienen:



Variante 5:

Das Programm wurde mit Kommentaren zu jeder Funktion und jeder Variable ausgestattet, um ein Lesen des Programms auch später problemlos zu ermöglichen.

Feld:

Keine bemerkenswerten Veränderungen vorgenommen.

Bienenstock:

Keine bemerkenswerten Veränderungen vorgenommen.

Biene:

Eine Biene hat nun die mögliche Eigenschaft "Sucher". Beim Erstellen des Bienenstocks wird einer jeden Biene im Stock – entsprechend einer übergebenen prozentuellen Zahl – die Eigenschaft Sucher mit true oder false zugeordnet.

Alle Bienen, die keine Sucher sind, werden gleichverteilt den Suchern zugewiesen.

Weiters haben die Bienen nun die 2D-Koordinate "BestZiel", welche jenen Punkt speichert, an dem die beste Nektarquelle gefunden wurde.

Außerdem wird nun ausgeschlossen, dass eine Biene beim 2D-Zufallsflug nur zwischen zwei Orten pendelt; d.h. der zuvor besuchte Ort kann von der Biene nicht mehr unmittelbar danach besucht werden (erst nach 3 Schritten ist dies wieder möglich).

Simulation:

Prinzipiell läuft die Simulation gleich wie in Variante 4, jedoch ist nun die Zusatzabfrage nach der Sucheigenschaft erforderlich.

Ist eine Biene ein Sucher, so sucht diese nach demselben Prinzip wie in Variante 4; zusätzlich werden nun beim Fund einer Blume ihre Koordinaten gespeichert, falls die Biene bisher keine bessere Quelle gefunden hat.

Ist eine Biene kein Sucher, so wird bei einer neuen Nektarsuche zuerst abgefragt, ob die zugehörige Sucherbiene eine Nektarquelle gefunden hat und ob bei dieser Quelle noch Nektar vorhanden ist; wenn ein solcher Ort existiert, so fliegt die Biene zu diesem Ort.

Wenn die oben Angegebenen Bedingungen nicht zutreffen, so startet die Biene eine Suche wie in Variante 4.

Diskussion der Variante:

Diese Art der Programmierung für den Tanz der Bienen entspricht nicht wirklich einem echten Bienenstock, da dort jede Biene ein Sucher sein kann. Das ist für unser Modell jedoch nicht so interessant, da das Hauptziel der Simulation die Informationsweitergabe ist, und diese so wie oben beschrieben durchgeführt wird.

Hauptsimulation:

Gewählte Parameter:

Geschwindigkeit der Biene: 20 [km/h]

Bienenanzahl zu Beginn: 20000 (~ Größe eines Bienenschwarms, Quelle: Brockhaus 2006)

Bienenwachstum: 500 [Stk/Tag] (Quelle: Brockhaus 2006 (maximal 1200))

Sucheraufteilung: 0 - 100% mit 10% Schrittweite

Immerblühende Pflanzen

Simulationsdauer: 10 Tage, Skalierung: 1/10

1. Simulation:

Abmessung des Feldes: 10 [km] Seitenlänge

Radius Schrittweite: 250 [m]

Trefferwahrscheinlichkeit: 5%, wenig Nektar pro Feld und schwacher Nachwuchs

Erste Pflanze:

Wahrscheinlichkeit: 1%

Maximale Nektarmenge pro [m²]: 5 []

Nektarnachwuchs pro [m²]: 1 []

Zweite Pflanze:

Wahrscheinlichkeit: 4%

Maximale Nektarmenge pro [m²]: 2 []

Nektarnachwuchs pro [m²]: 1 []

2. Simulation:

Abmessung des Feldes: 10 [km] Seitenlänge

Radius Schrittweite: 250 [m]

Trefferwahrscheinlichkeit: 2.5%, mittlere Nektarmenge und mittlerer Nachwuchs

Erste Pflanze:

Wahrscheinlichkeit: 0.5%

Maximale Nektarmenge pro [m²]: 10 []

Nektarnachwuchs pro [m²]: 2 []

Zweite Pflanze:

Wahrscheinlichkeit: 2%

Maximale Nektarmenge pro [m²]: 4 []

Nektarnachwuchs pro [m²]: 1 []

3. Simulation:

Abmessung des Feldes: 15 [km] Seitenlänge

Radius Schrittweite: 500 [m]

Trefferwahrscheinlichkeit: 1.25%, viel Nektar pro Feld und starker Nachwuchs

Erste Pflanze:

Wahrscheinlichkeit: 0.25%

Maximale Nektarmenge pro [m²]: 20 []

Nektarnachwuchs pro [m²]: 4 []

Zweite Pflanze:

Wahrscheinlichkeit: 1%

Maximale Nektarmenge pro [m²]: 8 []

Nektarnachwuchs pro [m²]: 2 []

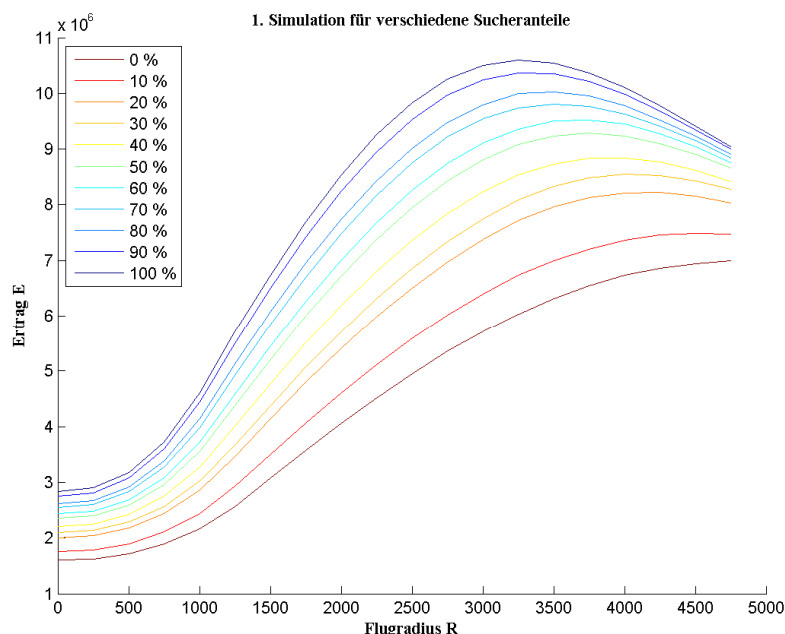
Diskussion der Ergebnisse:

Allgemein:

- Die Nektarmenge und dessen Nachwuchs können eigentlich völlig beliebig gewählt werden, da sie stark von den örtlichen Gegebenheiten und dem dortigen Pflanzenangebot abhängen (z.B. Gebirge oder Flachland). Deshalb sind diese Größen auch dimensionslos betrachtet worden.
- Die Daten zu Bienenanzahl, Bienennachwuchs und die Fluggeschwindigkeit stammen aus den oben angegebenen Quellen. Diese Werte sind jedoch nur Mittelwerte und können in Wirklichkeit auch mehr oder weniger davon abweichen. Der Bienennachwuchs wurde so gewählt, dass der Stock nach etwa 3 Monaten zu schwärmen beginnt bzw. sich teilt; dies stimmt mit den natürlichen Gegebenheiten überein.
- Das Intervall, in dem der Flugradius gewählt wurde, ist natürlich davon abhängig, wie groß das gesamte Feld ist. Da selbst mit C++ der Aufwand für die Simulation sehr groß wird, ist der gewählte Flugradius bei der ersten und zweiten Simulation auf 10km und bei der dritten auf 15km eingeschränkt worden.
- Im Folgenden ist erkennbar, dass der größte Ertrag jeweils bei 100% Sucheranteil erreicht wird; dies hängt wahrscheinlich damit zusammen, dass die empirisch gewählte Nektarmenge und Nektarverteilung zu gering sein wird.
- Weiters sieht man, dass der optimale Flugradius von der ersten zur dritten Simulation leicht ansteigt; das liegt daran, dass die Biene bei einer geringeren Wahrscheinlichkeit eine Blume zu finden einfach weiter fliegen muss, um eine Nektarquelle anzutreffen.

1. Simulation:

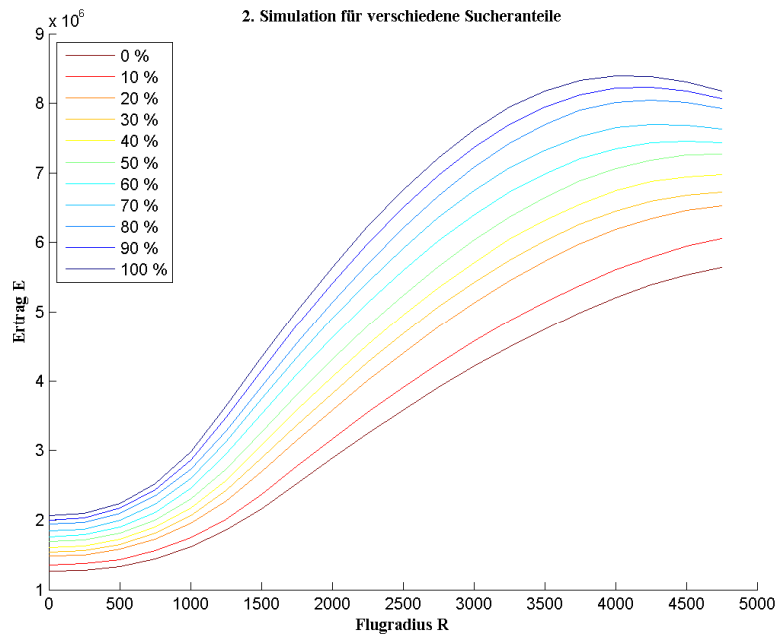
- Grafisches Ergebnis:



- Man erkennt, dass der maximale Nektarertrag bei 100% Sucheranteil liegt und der maximale Flugradius 3250m beträgt. Daraus folgt, dass der durchschnittliche Flugradius bei 2167m liegt (siehe Top-Down-Modell).

2. Simulation:

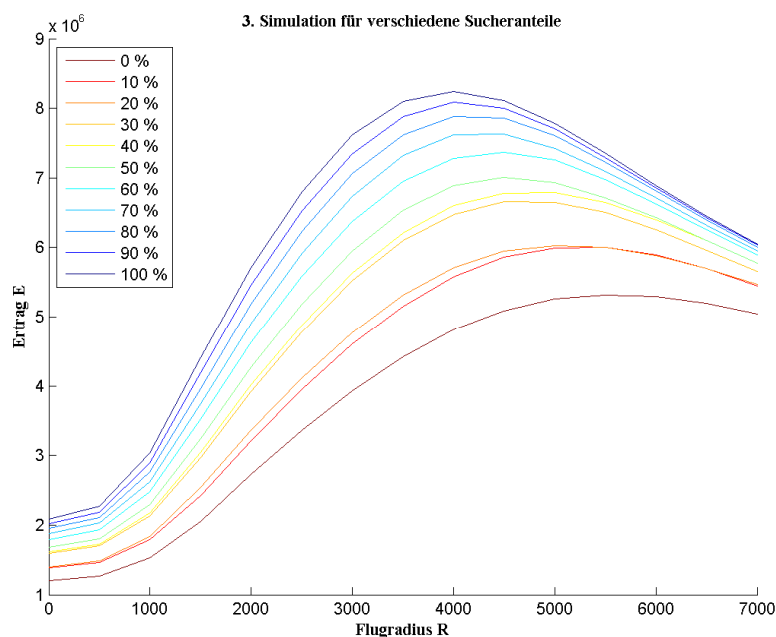
- Grafisches Ergebnis:



- Man erkennt, dass der maximale Nektarertrag wieder bei 100% Sucheranteil liegt, der maximale Flugradius jedoch 4000m beträgt. Daraus folgt, dass der durchschnittliche Flugradius hier bei 2667m liegt (siehe Top-Down-Modell).

3. Simulation:

- Grafisches Ergebnis:



- Man erkennt, dass das Ergebnis fast mit jenem der zweiten Simulation übereinstimmt. Das kann eventuell daran liegen, dass bei der dritten Simulation die radiale Auflösung 500m beträgt und deshalb das „echte“ Maximum um bis zu 250m vom ermittelten Wert abweichen kann.

Top-Down-Modellierung

Im Gegensatz zur Bottom-Up-Modellierung wird hier nicht jede einzelne Biene betrachtet, sondern nur die Gesamtheit der Bienen. Dies ist hier sinnvoll, da es bei 20000 Bienen praktisch egal ist, wie viel Nektar eine bestimmte Biene sammelt; entscheidend ist nur der Ertrag des gesamten Bienenstocks.

Außerdem wird das Modell ohne die in der Bottom-Up-Modellierung verwendete Suchereigenschaft einer Biene angesetzt, weshalb dieses auch nur qualitativ mit der Simulation übereinstimmt.

Verwendete Größen:

- | | |
|---|--|
| <ul style="list-style-type: none"> • N Anzahl an Bienen • T Arbeitszeit pro Tag • E_{ges} gesamter Nektarertrag • E Nektarertrag eines Kreisringes | <ul style="list-style-type: none"> • r maximaler Flugradius • v Fluggeschwindigkeit • q_w Nektarnachwuchs • q_B Tragleistung einer Biene |
|---|--|

Verwendetes Prinzip:

Der gesamte Nektarertrag, der vom kreisförmigen Feld eingebracht wird, ist gleich der Summe der Nektarerträge über kleinere Kreisringe.

Modell:

Der Nektarertrag eines (schmalen) Kreisringes mit Radius $\Delta\rho$ ist gleich dem Minimum folgender Größen:

- 1.) Nektarmenge, die auf dem Kreisring vorhanden ist bzw. nachwächst
- 2.) Nektarmenge, die von den Bienen zum Stock transportiert werden kann

Somit lässt sich der Ertrag nun folgendermaßen schreiben:

$$E(\rho) = \text{Min}\left(2 \cdot \pi \cdot \rho \cdot \Delta\rho \cdot q_w, T \cdot \frac{1}{2 \cdot \rho} \cdot N \cdot \frac{2 \cdot \pi \cdot \rho \cdot \Delta\rho}{r^2 \cdot \pi} \cdot q_B\right), \text{ wobei } N \cdot \frac{2 \cdot \pi \cdot \rho \cdot \Delta\rho}{r^2 \cdot \pi} \text{ die}$$

v

Anzahl an Bienen ist, die zum Kreisring mit Radius ρ fliegen.

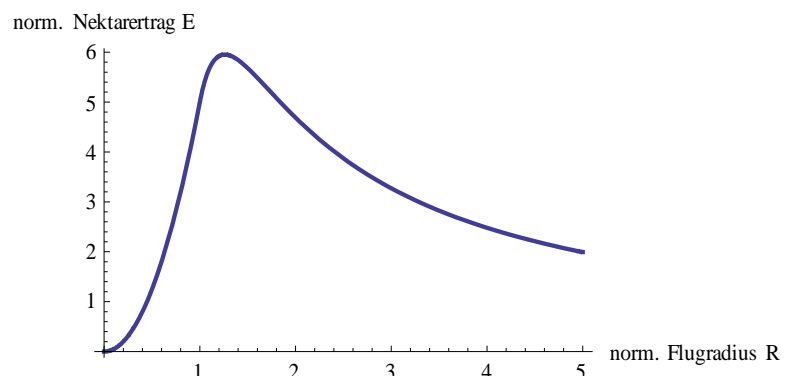
Fasst man die vorkommenden Konstanten nun zu $k_1 = 2 \cdot \pi \cdot q_w$ und $k_2 = T \cdot N \cdot v \cdot q_B$ zusammen, so vereinfacht sich der Ausdruck für $E(\rho)$ und man erhält:

$$E(\rho) = \text{Min}\left(k_1 \cdot \rho, \frac{k_2}{r^2}\right) \cdot \Delta\rho$$

Der Ertrag des gesamten kreisförmigen Feldes ist dann die Summe dieser Einzelerträge bzw. für $\Delta\rho \rightarrow 0$ das Integral über diese:

$$E_{ges}(r) = \int_0^r \text{Min}\left(k_1 \cdot \rho, \frac{k_2}{r^2}\right) \cdot d\rho$$

Dieses Integral kann z.B. mit MATHEMATICA für verschiedene Werte r numerisch berechnet und grafisch dargestellt werden. Man erkennt deutlich die Analogien zum Ergebnis der Simulation.



Bemerkungen:

- 1.) Sowohl aus der Simulation als auch aus der theoretischen Betrachtung erkennt man, dass für kleines r der Nektarertrag quadratisch mit dem Flugradius zunimmt, während er für großes r mit $\frac{1}{r}$ abnimmt; dazwischen – beim optimalen (maximalen) Flugradius – liegt ein deutliches Maximum.
- 2.) Bisher wurde immer nur vom maximalen Flugradius der Bienen gesprochen; interessanter wäre aber der durchschnittliche Radius, den die Bienen fliegen. Dieser ergibt sich aus folgender Überlegung:

$$\bar{r} = \frac{\int_0^r \rho \cdot 2 \cdot \pi \cdot \rho \cdot d\rho}{r^2 \cdot \pi} = \frac{2 \cdot \pi \cdot \int_0^r \rho^2 \cdot d\rho}{r^2 \cdot \pi} = \frac{2 \cdot \pi \cdot \frac{r^3}{3}}{r^2 \cdot \pi} = \frac{2}{3} \cdot r$$

- 3.) Der in der Natur beobachtete durchschnittliche Flugradius der Bienen beträgt 1 bis 2 Kilometer. Das von uns ermittelte Ergebnis von 2 bis 3 Kilometer deutet darauf hin, dass die Nektarmenge wahrscheinlich zu gering gewählt wurde.