

# Simulation normalisierter BPD L und TSS Prozesse

## 2. Vortrag des Begleitseminars

Boris Prochnau

Institut für Angewandte Mathematik  
Universität Bonn

16. Juni 2014

# Übersicht

- ① Ziel
- ② Model  
Normalisierung  
Equilibrium
- ③ Algorithmus
- ④ Simulation  
Aufgabenteilung und Flexibilität  
Layout
- ⑤ Korrektheit der Implementation
- ⑥ TSS - Prozesse  
Fitness  
Interpolation

# Ziel der Bachelorarbeit

- Simulation eines normalisierten BPDFL Prozesses
- Simulation eines normalisierten TSS Prozesses

# Grundlagen - Wiederholung

- Jedes Individuum hat ein Merkmal  $x \in X$ .  
Der Einfachheit halber sei  $X$  eine Indexmenge:  
 $X = \{1, \dots, n\}$  repräsentativ für eine Durchzählung der  
Merkmale.

# Grundlagen - Wiederholung

- Jedes Individuum hat ein Merkmal  $x \in X$ .  
Der Einfachheit halber sei  $X$  eine Indexmenge:  
 $X = \{1, \dots, n\}$  repräsentativ für eine Durchzählung der Merkmale.
- Jedes Individuum kann sich asexuell fortpflanzen oder sterben

# Grundlagen - Wiederholung

- Jedes Individuum hat ein Merkmal  $x \in X$ .  
Der Einfachheit halber sei  $X$  eine Indexmenge:  
 $X = \{1, \dots, n\}$  repräsentativ für eine Durchzählung der Merkmale.
- Jedes Individuum kann sich asexuell fortpflanzen oder sterben
- Diese Ereignisse sind exponentiell verteilte Zeitpunkte.

# Raten

Mit folgenden Raten:

- $b(x)$ : Geburtenraten durch ein Individuum mit Merkmal  $x$

# Raten

Mit folgenden Raten:

- $b(x)$ : Geburtenraten durch ein Individuum mit Merkmal  $x$
- $d(x)$ : natürliche Todesrate



## Raten

Mit folgenden Raten:

- $b(x)$ : Geburtenraten durch ein Individuum mit Merkmal  $x$
- $d(x)$ : natürliche Todesrate
- $c(x,y)$ : Todesrate durch Wettbewerb zwischen Individuen mit Merkmal  $x$  und  $y$ .

## Raten

Mit folgenden Raten:

- $b(x)$ : Geburtenraten durch ein Individuum mit Merkmal  $x$
- $d(x)$ : natürliche Todesrate
- $c(x,y)$ : Todesrate durch Wettbewerb zwischen Individuen mit Merkmal  $x$  und  $y$ .
- $\mu$ : Mutationswahrscheinlichkeit "auf die Nachbarn" mit je  $\frac{\mu}{2}$  pro Nachbar.

# kompakte Raten - Superposition

Zusammenfassen der Ereignisse ergibt:

- intrinsische Geburtenrate:  $b(x) \cdot (1 - \mu)$

# kompakte Raten - Superposition

Zusammenfassen der Ereignisse ergibt:

- intrinsische Geburtenrate:  $b(x) \cdot (1 - \mu)$
- Todesrate:  $d(x) + \sum_{i=1}^{N_t} c(x, x_i)$ ,  $N_t = \# \text{Individuen}$  zum Zeitpunkt  $t$ , und  $x_i$  das Merkmal des  $i$ -ten Individuums.

## kompakte Raten - Superposition

Zusammenfassen der Ereignisse ergibt:

- intrinsische Geburtenrate:  $b(x) \cdot (1 - \mu)$
- Todesrate:  $d(x) + \sum_{i=1}^{N_t} c(x, x_i)$ ,  $N_t = \# \text{Individuen}$  zum Zeitpunkt  $t$ , und  $x_i$  das Merkmal des  $i$ -ten Individuums.
- ODER Todesrate:  $d(x) + \sum_{i=1}^n c(x, x_i) \cdot n_t(x_i)$ ,  $n = \# \text{Merkmale}$ , und  $n_t(x_i) = \# \text{Individuen}$  mit Merkmal  $x_i$  zur Zeit  $t$ .

## Merkmale im Blickpunkt

Programm soll Entwicklung der Merkmale simulieren, nicht der Individuen:

## Merkmale im Blickpunkt

Programm soll Entwicklung der Merkmale simulieren, nicht der Individuen:

- Geburtenrate (Wachstumsrate) des Merkmals  $x$ :

$$\begin{aligned}
 B(x) = & b(x) \cdot (1 - \mu) \cdot n_t(x) \\
 & + \frac{\mu}{2} \cdot b(x+1) \cdot n_t(x+1) \\
 & + \frac{\mu}{2} \cdot b(x-1) \cdot n_t(x-1)
 \end{aligned}$$

## Merkmale im Blickpunkt

Programm soll Entwicklung der Merkmale simulieren, nicht der Individuen:

- Geburtenrate (Wachstumsrate) des Merkmals  $x$ :

$$\begin{aligned} B(x) &= b(x) \cdot (1 - \mu) \cdot n_t(x) \\ &\quad + \frac{\mu}{2} \cdot b(x+1) \cdot n_t(x+1) \\ &\quad + \frac{\mu}{2} \cdot b(x-1) \cdot n_t(x-1) \end{aligned}$$

- Todesrate des Merkmals  $x$ :

$$D(x) = d(x) \cdot n_t(x) + n_t(x) \cdot \sum_{i=1}^n c(x, x_i) \cdot n_t(x_i)$$



## Merkmale im Blickpunkt

Programm soll Entwicklung der Merkmale simulieren, nicht der Individuen:

- Geburtenrate (Wachstumsrate) des Merkmals  $x$ :

$$\begin{aligned} B(x) &= b(x) \cdot (1 - \mu) \cdot n_t(x) \\ &\quad + \frac{\mu}{2} \cdot b(x+1) \cdot n_t(x+1) \\ &\quad + \frac{\mu}{2} \cdot b(x-1) \cdot n_t(x-1) \end{aligned}$$

- Todesrate des Merkmals  $x$ :

$$D(x) = d(x) \cdot n_t(x) + n_t(x) \cdot \sum_{i=1}^n c(x, x_i) \cdot n_t(x_i)$$

⇒ 2 exponentiellen Uhren (Zeitpunkten) pro Merkmal

# Totale Ereignis Rate

Zusammenfassung zu einer Uhr pro Evolutionssprung:

# Totale Ereignis Rate

Zusammenfassung zu einer Uhr pro Evolutionssprung:

- Ereignisrate des Merkmals  $x$  (Trait Rate):

$$TR(x) = B(x) + D(x)$$

- Totale Ereignis Rate (Total Event Rate):

$$TER = \sum_{x \in X} TR(x)$$

Die TER entspricht einer Rate für das erste klingeln der Merkmale.

# Population als Zufallsvariable - Wiederholung

# Population als Zufallsvariable - Wiederholung

Die Population ist ein Markov Sprungprozess der durch Zufallsvariablen

$$\nu_t = \sum_{i=1}^{N_t} \delta_{x_i}, \text{ mit } \int_X 1 \nu_t(dx) = N_t$$

beschrieben wird.

Wobei:

$$\nu_t \in M_F(X) = \left\{ \sum_{i=1}^n \delta_{x_i}, n \in \mathbb{N}, x_1, \dots, x_n \in X \right\}$$

# Large Population Approximation

Die LPA Normalisierung erweitert die Betrachtung auf die Ebene der Population.

# Large Population Approximation

Die LPA Normalisierung erweitert die Betrachtung auf die Ebene der Population. Dafür wird der Prozess mit einem Parameter  $K$  skaliert:

$$\nu_t^K := \frac{1}{K} \nu_t$$

# Large Population Approximation

Die LPA Normalisierung erweitert die Betrachtung auf die Ebene der Population. Dafür wird der Prozess mit einem Parameter  $K$  skaliert:

$$\nu_t^K := \frac{1}{K} \nu_t$$

Mit Anpassungen:

- $n_0^K$  wird proportional zu  $K$  gewählt



# Large Population Approximation

Die LPA Normalisierung erweitert die Betrachtung auf die Ebene der Population. Dafür wird der Prozess mit einem Parameter  $K$  skaliert:

$$\nu_t^K := \frac{1}{K} \nu_t$$

Mit Anpassungen:

- $n_0^K$  wird proportional zu  $K$  gewählt
- Raten für Geburten und natürliche Tode der Individuen bleiben unverändert

# Large Population Approximation

Die LPA Normalisierung erweitert die Betrachtung auf die Ebene der Population. Dafür wird der Prozess mit einem Parameter  $K$  skaliert:

$$\nu_t^K := \frac{1}{K} \nu_t$$

Mit Anpassungen:

- $n_0^K$  wird proportional zu  $K$  gewählt
- Raten für Geburten und natürliche Tode der Individuen bleiben unverändert
- Jedoch:  $c^K = \frac{c}{K}$

# Large Population Approximation

Die LPA Normalisierung erweitert die Betrachtung auf die Ebene der Population. Dafür wird der Prozess mit einem Parameter  $K$  skaliert:

$$\nu_t^K := \frac{1}{K} \nu_t$$

Mit Anpassungen:

- $n_0^K$  wird proportional zu  $K$  gewählt
- Raten für Geburten und natürliche Tode der Individuen bleiben unverändert
- Jedoch:  $c^K = \frac{c}{K}$
- proportionale Anpassung von  $\mu$

## Beispiel: $K = 100$

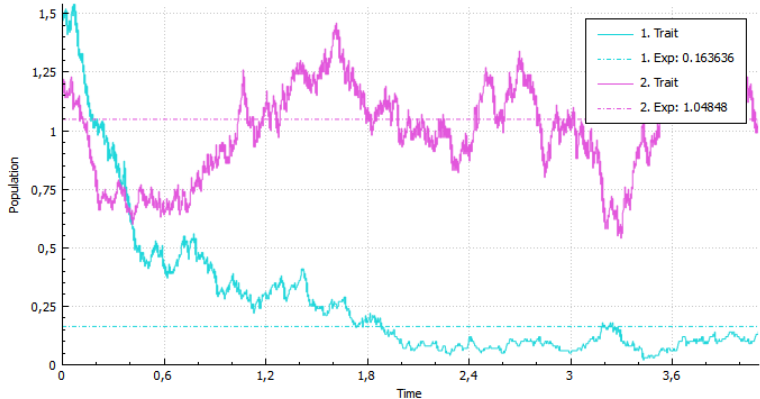


Abbildung: LPA Normalisierung mit  $K=100$

# Beispiel: $K = 10000$

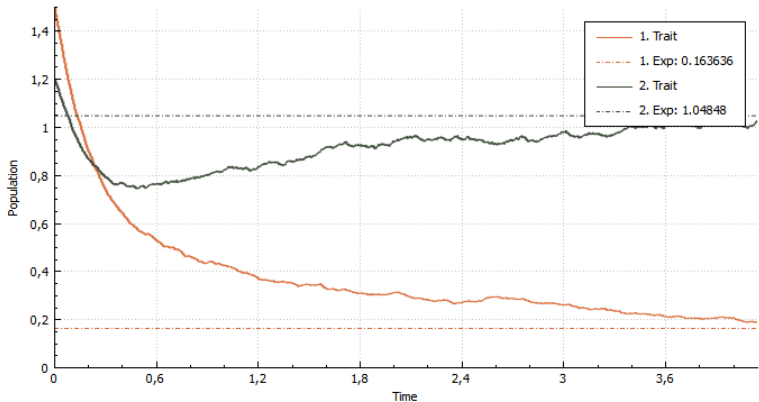


Abbildung: LPA Normalisierung mit  $K=10000$

## stabile Zustände

Im stabilen Zustand ändert sich die Populationsgröße nicht mehr:

## stabile Zustände

Im stabilen Zustand ändert sich die Populationsgröße nicht mehr:

- Für Monomorphe Population:

$$0 = \dot{n} = (b(x) - d(x) - \bar{n}c(x, x))\bar{n}$$
$$\bar{n}_x = \frac{[b(x) - d(x)]_+}{c(x, x)}$$

## stabile Zustände

Im stabilen Zustand ändert sich die Populationsgröße nicht mehr:

- Für Monomorphe Population:

$$0 = \dot{n} = (b(x) - d(x) - \bar{n}c(x, x))\bar{n}$$

$$\bar{n}_x = \frac{[b(x) - d(x)]_+}{c(x, x)}$$

- Für Dimorphe Population:

$$n_x = \frac{(b(x) - d(x))c(y, y) - (b(y) - d(y))c(x, y)}{c(y, y)c(x, x) - c(y, x)c(x, y)}$$

oder  $(\bar{n}_x, 0)$ ,  $(0, \bar{n}_y)$  bzw.  $(0, 0)$



## Evolution Step

Der Simulation liegt ein Algorithmus zugrunde der einen Sprung des Markov Sprung Prozesses durchführt.

## Evolution Step

Der Simulation liegt ein Algorithmus zugrunde der einen Sprung des Markov Sprung Prozesses durchführt.

---

**Algorithm 2** EvolutionStep()
 

---

**Ensure:** A full evolution Step happened

- 1: calculateEventRates();
  - 2: sampleEventTime();
  - 3: changeATrait();
-

## Evolution Step - detaillierter

Von dieser werden folgende Berechnungen angestoßen:

## Evolution Step - detaillierter

Von dieser werden folgende Berechnungen angestoßen:

---

### Algorithm 4 EvolutionStep()

---

**Ensure:** A full evolution Step happened

- 1: —>calculateEventRates();
  - 2: calculateTotalDeathRates()
  - 3: calculateTotalBirthRates()
  - 4: calculateTotalEventRate()
  - 5: —>sampleEventTime();
  - 6: sampleEventTime();
  - 7: —>changeATrait();
  - 8: choseTraitToChange();
  - 9: choseEventType();
  - 10: executeEventTypeOnTrait();
-

# Flexibilität

Um Flexibilität aufrecht zu erhalten werden die Arbeitsbereiche  
im Code getrennt gehalten

# Flexibilität

Um Flexibilität aufrecht zu erhalten werden die Arbeitsbereiche im Code getrennt gehalten

Grund der Idee:

- Möglichst viel Unabhängigkeit

# Flexibilität

Um Flexibilität aufrecht zu erhalten werden die Arbeitsbereiche im Code getrennt gehalten

Grund der Idee:

- Möglichst viel Unabhängigkeit
- verhindert "Coderot faulen Code"

# Flexibilität

Um Flexibilität aufrecht zu erhalten werden die Arbeitsbereiche im Code getrennt gehalten

Grund der Idee:

- Möglichst viel Unabhängigkeit
- verhindert "Coderot faulen Code"
- steigende Komplexität führt nicht zu undefiniertem Verhalten



# Flexibilität

Um Flexibilität aufrecht zu erhalten werden die Arbeitsbereiche im Code getrennt gehalten

Grund der Idee:

- Möglichst viel Unabhängigkeit
- verhindert "Coderot faulen Code"
- steigende Komplexität führt nicht zu undefiniertem Verhalten
- "Single Responsibility Principle"

# Flexibilität

Um Flexibilität aufrecht zu erhalten werden die Arbeitsbereiche im Code getrennt gehalten

Grund der Idee:

- Möglichst viel Unabhängigkeit
- verhindert "Coderot faulen Code"
- steigende Komplexität führt nicht zu undefiniertem Verhalten
- "Single Responsibility Principle"
- Keine Klassen die zu viel Wissen

# Arbeitsmodule

Die Architektur besteht aus 3 Modulen

## Arbeitsmodule

Die Architektur besteht aus 3 Modulen

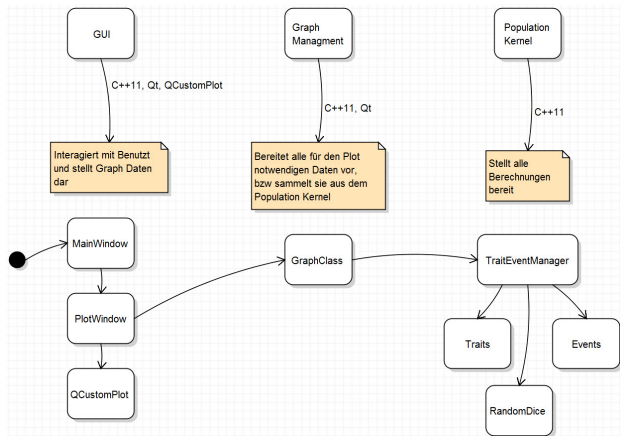


Abbildung: Arbeitsmodule und Klassenabhängigkeiten

## Layout: Lesen der Parameter

Parameter sollten aus Dateien gelesen werden können:

## Layout: Lesen der Parameter

Parameter sollten aus Dateien gelesen werden können:

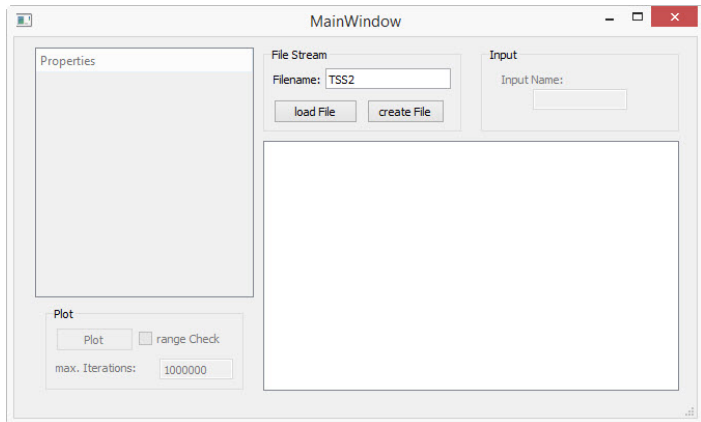


Abbildung: MainWindow nach dem Start

# Anzeige der Parameter

Baumdarstellung der Parameter:

## Anzeige der Parameter

Baumdarstellung der Parameter:

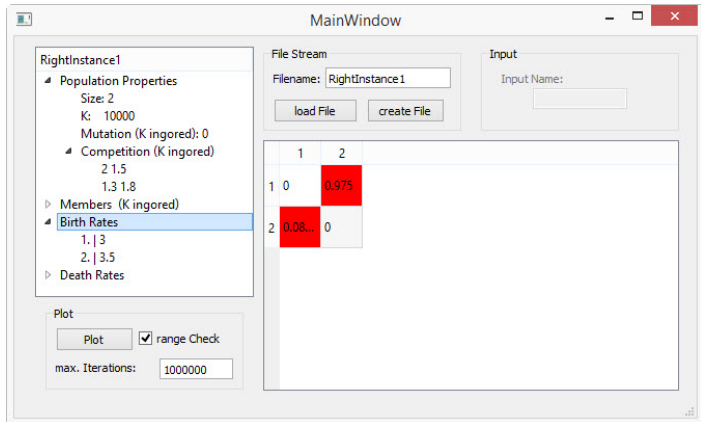


Abbildung: MainWindow mit geladenen Parametern

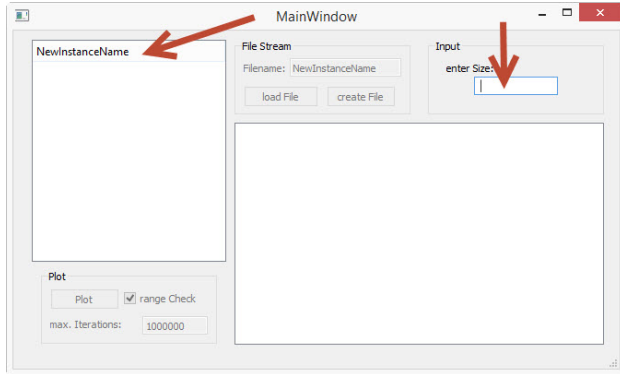


# Erstellen neuer Testinstanzen

Anlegen einer neuen Datei:

## Erstellen neuer Testinstanzen

Anlegen einer neuen Datei:



**Abbildung:** Nach Klick auf "create File" werden die neuen Parameter einzeln abgefragt

# Testinstanz erstellt

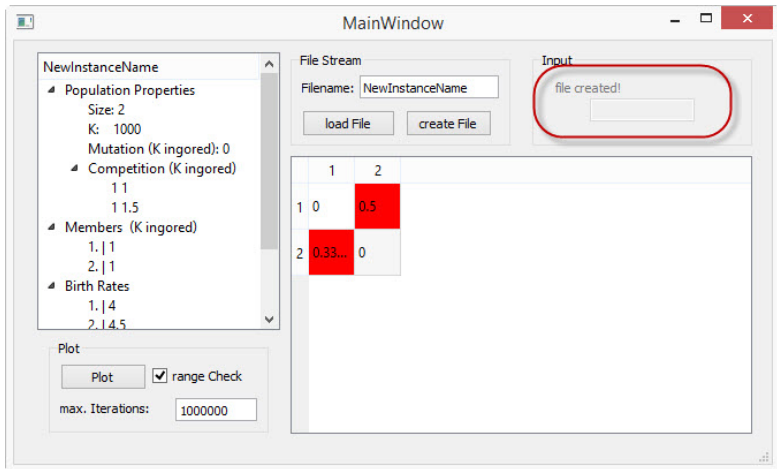


Abbildung: Nach eingabe des letzten Parameters

# Darstellung der Graphen

Was soll die graphische Darstellung der Graphen erfüllen?

# Darstellung der Graphen

Was soll die graphische Darstellung der Graphen erfüllen?

- Anzeige des simulierten Prozesses

# Darstellung der Graphen

Was soll die graphische Darstellung der Graphen erfüllen?

- Anzeige des simulierten Prozesses
- Zoom und Bewegung auf einem Koordinatensystem

# Darstellung der Graphen

Was soll die graphische Darstellung der Graphen erfüllen?

- Anzeige des simulierten Prozesses
- Zoom und Bewegung auf einem Koordinatensystem
- Abspeichern aktueller Bilder für spätere Vergleiche

# Darstellung der Graphen

Was soll die graphische Darstellung der Graphen erfüllen?

- Anzeige des simulierten Prozesses
- Zoom und Bewegung auf einem Koordinatensystem
- Abspeichern aktueller Bilder für spätere Vergleiche
- Verhindern dass die Berechnung das Programm einfriert



# Start der Darstellung

Nach dem drücken des "Plot" Buttons öffnet sich ein Fenster

# Start der Darstellung

Nach dem drücken des "Plot" Buttons öffnet sich ein Fenster

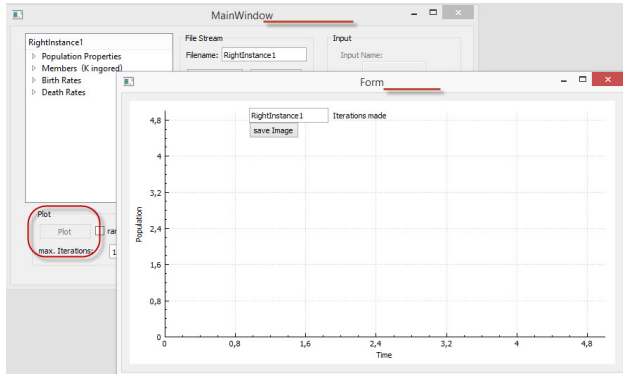


Abbildung: Start des PlotWindow

## Arbeit im Hintergrund

Was fällt auf?

- Der Plot Button kann nicht mehr betätigt werden. Das verdeutlicht, dass der Prozess gerade simuliert wird

## Arbeit im Hintergrund

Was fällt auf?

- Der Plot Button kann nicht mehr betätigt werden. Das verdeutlicht, dass der Prozess gerade simuliert wird
- Trotz der Berechnungen friert das Bild nicht ein und verursacht keinen Konflikt mit der Betriebssystem-Sicherheit:

## Arbeit im Hintergrund

Was fällt auf?

- Der Plot Button kann nicht mehr betätigt werden. Das verdeutlicht, dass der Prozess gerade simuliert wird
- Trotz der Berechnungen friert das Bild nicht ein und verursacht keinen Konflikt mit der Betriebssystem-Sicherheit:

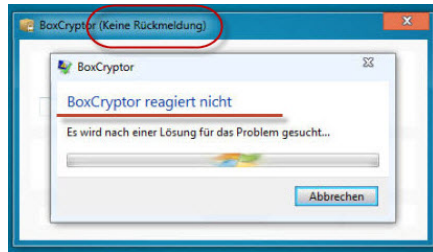


Abbildung: Bsp: Überlasteter Hauptthread

## Darstellung der Graphen

Wenn ein günstiger Zustand erreicht wurde, oder maximal viele Iterationen gemacht wurden, werden die Punkte verbunden:

# Darstellung der Graphen

Wenn ein günstiger Zustand erreicht wurde, oder maximal viele Iterationen gemacht wurden, werden die Punkte verbunden:

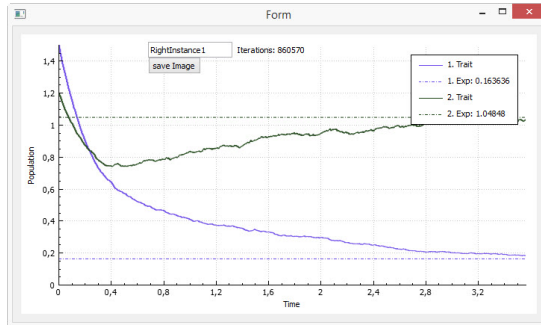


Abbildung: PlotWindow mit Dimorpher Population

# Was wurde Dargestellt?



# Was wurde Dargestellt?

- Die Entwicklung der beiden Merkmale mit Zeit und Größe
- Die stabilen Zustände (gestrichelt)
- Die Anzahl der tatsächlich gemachten Sprünge
- Einen Button zum Speichern des Bildes
- Eine Legende die jeden Graphen beschreibt

# Zoom und Bewegungsfreiheit

Zoom, Bewegungsfreiheit und Reskalierung des Plots sind auch möglich:

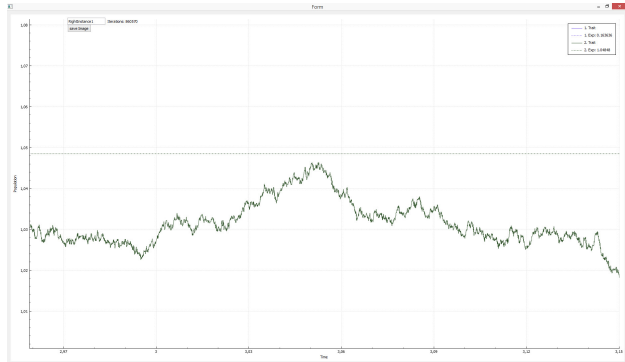


Abbildung: Plot wurde maximiert, gezoomt und bewegt

Simulation  
normalisier-  
ter  
Prozesse

B.Prochnau

Ziel

Model

Normalisierung  
Equilibrium

Algorithmus

Simulation

Aufgabenteilung  
und  
Flexibilität  
Layout

Korrektheit  
der Imple-  
mentation

TSS -

Prozesse

Fitness  
Interpolation

# Korrektheit: Testgetriebenen Entwicklung

# Korrektheit: Testgetriebenen Entwicklung

- Die Korrektheit der Implementation ist mit zunehmender Komplexität des Codes schwerer zu prüfen (besonders bei Zufallsbedingten Simulationen)

# Korrektheit: Testgetriebenen Entwicklung

- Die Korrektheit der Implementation ist mit zunehmender Komplexität des Codes schwerer zu prüfen (besonders bei Zufallsbedingten Simulationen)
- Zu diesem Zweck verwende ich das Prinzip der "Testgetriebenen Entwicklung"(Test Driven Development)

# Korrektheit: Testgetriebenen Entwicklung

- Die Korrektheit der Implementation ist mit zunehmender Komplexität des Codes schwerer zu prüfen (besonders bei Zufallsbedingten Simulationen)
- Zu diesem Zweck verwende ich das Prinzip der "Testgetriebenen Entwicklung"(Test Driven Development)
- Dabei werden Funktionen mit erwartetem Verhalten verglichen

# Korrektheit: Testgetriebenen Entwicklung

- Die Korrektheit der Implementation ist mit zunehmender Komplexität des Codes schwerer zu prüfen (besonders bei Zufallsbedingten Simulationen)
- Zu diesem Zweck verwende ich das Prinzip der "Testgetriebenen Entwicklung"(Test Driven Development)
- Dabei werden Funktionen mit erwartetem Verhalten verglichen

Folgend ein Beispiel:

## Einfaches Testbeispiel

```

51 void TraitEventManagerTest::verifyWrittenData()
52 {
53     QCOMPARE(TraitClass::Size, 3.);
54     QCOMPARE(TraitClass::Mutation, 0.1);
55     for(int i = 0; i < TraitClass::Size; ++i){
56         QCOMPARE(Manager.Trait[i].BirthRate, 10.);
57         QCOMPARE(Manager.Trait[i].DeathRate, 5.);
58         QCOMPARE(TraitClass::CompDeathRate[i][i], 2.);
59     }
60 }
61
62 // ----- section 1: Rates -----
63 /// Unit Tests for INPUT VALIDATION
64
65 void TraitEventManagerTest::readAndClearStandardInput()
66 {
67     Manager.initWithFile("ValidateTests.txt");
68     verifyWrittenData();
69     Manager.clearData();
70     QVERIFY(TraitClass::Size == 0.);
71     QVERIFY(Manager.Trait.size() == 0.);
72     QVERIFY(TraitClass::CompDeathRate.size() == 0.);
73     QVERIFY(Manager.Trait.size() == 0.);
74 }
75

```

Abbildung: Unit Test versichert korrektes lesen aus Datei



# Testdurchlauf

Der Output einer Testsammlung kann so beginnen:

## Testdurchlauf

Der Output einer Testsammlung kann so beginnen:

```

Ausgabe der Anwendung
TraitEventManager x
Starte D:\Thesis\FinalRegulatedPopulation\build-TraitEventManager-Desktop_Qt_5_2_1_MinGW_32bit-
***** Start testing of TraitEventManagerTest *****
Config: Using QTest library 5.2.1, Qt 5.2.1
PASS : TraitEventManagerTest::initTestCase()
PASS : TraitEventManagerTest::readAndClearStandardInput()
PASS : TraitEventManagerTest::verifyTotalIntrinsicDeathRate()
PASS : TraitEventManagerTest::verifyTotalCompDeathRate()
QDEBUG : TraitEventManagerTest::verifyTotalDeathRate() verify total death rates ...
QDEBUG : TraitEventManagerTest::verifyTotalDeathRate() trait 0 total death rate: 30500
QDEBUG : TraitEventManagerTest::verifyTotalDeathRate() trait 1 total death rate: 25500
QDEBUG : TraitEventManagerTest::verifyTotalDeathRate() trait 2 total death rate: 40500
PASS : TraitEventManagerTest::verifyTotalDeathRate()
QDEBUG : TraitEventManagerTest::verifyTotalBirthRate() verify total birth rates ...
QDEBUG : TraitEventManagerTest::verifyTotalBirthRate() trait 0 total birth rate: 1050 verified
QDEBUG : TraitEventManagerTest::verifyTotalBirthRate() trait 1 total birth rate: 1100 verified
QDEBUG : TraitEventManagerTest::verifyTotalBirthRate() trait 2 total birth rate: 1050 verified
PASS : TraitEventManagerTest::verifyTotalBirthRate()
QDEBUG : TraitEventManagerTest::verifyEventRates() verify: Total Event Rate = 99700 ...
  
```

Abbildung: Ergebnisse einiger Tests

## Testdurchlauf

Der Output einer Testsammlung kann so beginnen:

```

Ausgabe der Anwendung
TraitEventManager
Starte D:\Thesis\FinalRegulatedPopulation\build-TraitEventManager-Desktop_Qt_5_2_1_MinGW_32bit-
***** Start testing of TraitEventManagerTest *****
Config: Using QTest library 5.2.1, Qt 5.2.1
PASS : TraitEventManagerTest::initTestCase()
PASS : TraitEventManagerTest::readAndClearStandardInput()
PASS : TraitEventManagerTest::verifyTotalIntrinsicDeathRate()
PASS : TraitEventManagerTest::verifyTotalCompDeathRate()
QDEBUG : TraitEventManagerTest::verifyTotalDeathRate() verify total death rates ...
QDEBUG : TraitEventManagerTest::verifyTotalDeathRate() trait 0 total death rate: 30500
QDEBUG : TraitEventManagerTest::verifyTotalDeathRate() trait 1 total death rate: 25500
QDEBUG : TraitEventManagerTest::verifyTotalDeathRate() trait 2 total death rate: 40500
PASS : TraitEventManagerTest::verifyTotalDeathRate()
QDEBUG : TraitEventManagerTest::verifyTotalBirthRate() verify total birth rates ...
QDEBUG : TraitEventManagerTest::verifyTotalBirthRate() trait 0 total birth rate: 1050 verified
QDEBUG : TraitEventManagerTest::verifyTotalBirthRate() trait 1 total birth rate: 1100 verified
QDEBUG : TraitEventManagerTest::verifyTotalBirthRate() trait 2 total birth rate: 1050 verified
PASS : TraitEventManagerTest::verifyTotalBirthRate()
QDEBUG : TraitEventManagerTest::verifyEventRates() verify: Total Event Rate = 99700 ...
  
```

Abbildung: Ergebnisse einiger Tests

Tests ermöglichen zusätzlich komplexere Simulationen

# Trait Substitution Sequence

- Wie bei LPA-Normalisierung ergeben sich TSS-Prozesse als Grenzprozesse von BPDF-Prozessen
- Jedoch mit wachsendem  $K$  schrumpft  $\mu$  mit der Ordnung:

$$\frac{1}{e^{\sqrt{K}}} \ll \mu_K \ll \frac{1}{K \log(K)}$$

- Weiterhin wird die Zeit skaliert so dass die Verdrängungszeit infinitesimal klein wird
- Für die Simulation bedeutet es, dass sehr viele Sprünge um das Equilibrium zu erwarten sind

# Bisheriger Simulationsstand

Eine Simulation würde bisher so aussehen ( $K = 1000$ ):

## Bisheriger Simulationsstand

Eine Simulation würde bisher so aussehen ( $K = 1000$ ):

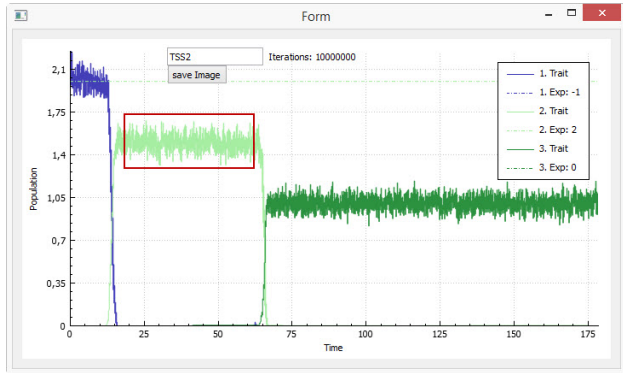


Abbildung: TSS Prozess mit  $K=1000$

Simulation  
normalisier-  
ter  
Prozesse

B.Prochnau

Ziel

Model

Normalisierung  
Equilibrium

Algorithmus

Simulation

Aufgabenteilung  
und  
Flexibilität  
Layout

Korrektheit  
der Imple-  
mentation

TSS -

Prozesse

**Fitness**

Interpolation

# Fitness-Funktion

# Fitness-Funktion

$$f(x, y) = b(x) - d(x) - c(x, y)\bar{n}_y$$



# Fitness-Funktion

$$f(x, y) = b(x) - d(x) - c(x, y)\bar{n}_y$$

- Sie gibt an wie gut sich ein Merkmal durchsetzen kann
- Asymptotische Wachstumsrate von  $y$ , wenn  $x$  im Zustand  $\bar{n}_x$  ist und  $y$  nur wenige Individuen hat
- Ermöglicht Aussagen über die Überlebenswahrscheinlichkeit einer Mutation
- Ermöglicht Aussagen über die angenommenen stabilen Zustände.

# Fitness-Funktion

$$f(x, y) = b(x) - d(x) - c(x, y)\bar{n}_y$$

- Sie gibt an wie gut sich ein Merkmal durchsetzen kann
- Asymptotische Wachstumsrate von  $y$ , wenn  $x$  im Zustand  $\bar{n}_x$  ist und  $y$  nur wenige Individuen hat
- Ermöglicht Aussagen über die Überlebenswahrscheinlichkeit einer Mutation
- Ermöglicht Aussagen über die angenommenen stabilen Zustände.

Da wir nur eine Mutation zu den Nachbarn berücksichtigen, ist unsere Fitness Matrix eine Bandmatrix

# Fitness-Matrix

Fitness-Matrix wird sofort beim Laden der Parameter berechnet und angezeigt:

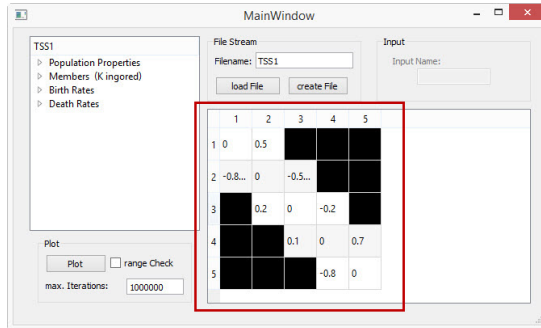


Abbildung: Fitness Bandmatrix

# Eigenschaften der Fitness

Was erwartet die Simulation von der Fitness?

Im dimorphen Fall gilt:

- $f(y, x) < 0 \Rightarrow (\bar{n}_x, 0)$  ist ein stabiler Zustand
- $f(y, x) > 0 \wedge f(x, y) < 0 \Rightarrow (\bar{n}_x, 0)$  ist ein stabiler Zustand

Für TSS-Prozesse gilt:

- $f(y, x) > 0 \wedge f(x, y) < 0$ , x wird durch y verdrängt
- $f(y, x) > 0 \wedge f(x, y) > 0$ , Koexistenz

# Invasionswahrscheinlichkeit

Die Fitness ermöglicht eine Grenzwertaussage über die Invasionswahrscheinlichkeit:

$$\frac{[f(y, x)]_+}{b(y)}$$

# Invasionswahrscheinlichkeit

Die Fitness ermöglicht eine Grenzwertaussage über die Invasionswahrscheinlichkeit:

$$\frac{[f(y, x)]_+}{b(y)}$$

Mit diesen Informationen lässt sich die Anzeige der Fitnessmatrix mit mehr Optionen ausstatten:

# Invasionswahrscheinlichkeit

Die Fitness ermöglicht eine Grenzwertaussage über die Invasionswahrscheinlichkeit:

$$\frac{[f(y, x)]_+}{b(y)}$$

Mit diesen Informationen lässt sich die Anzeige der Fitnessmatrix mit mehr Optionen ausstatten:  
Einträge werden:

- Rot - falls eine Koexistenz von Merkmalen zu erwarten ist
- Grün - falls die Invasionswahrscheinlichkeit hoch ist

Geplant ist eine stufenweiser Anstieg von hellem zu dunklem Grün. Wurde noch nicht implementiert.

# Fitnessmatrix mit farblichen Akzenten

Hier sieht man eine Fitnessmatrix mit grünen und roten Einträgen. Dabei wird ein Eintrag grün wenn er eine Invasionswahrscheinlichkeit von mindestens 50% aufweist.

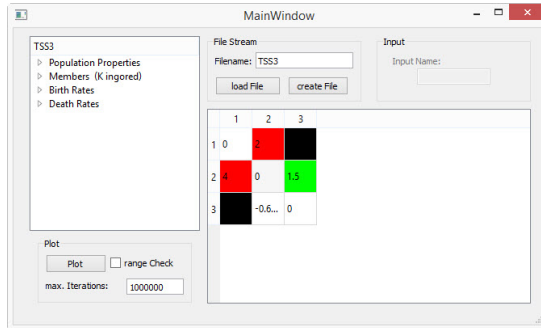


Abbildung: Fitness Matrix mit roten und gruenen akzenten



# Optimierung

Um die Simulationsdauer zu reduzieren würde sich eine lineare Interpolation des Prozesses anbieten. Die rechnerische Entlastung wird im folgenden Bild deutlich dargestellt:

# Optimierung

Um die Simulationsdauer zu reduzieren würde sich eine lineare Interpolation des Prozesses anbieten. Die rechnerische Entlastung wird im folgenden Bild deutlich dargestellt:

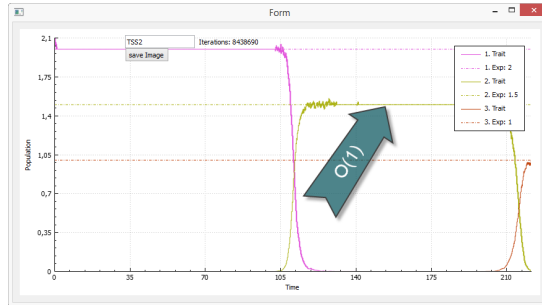


Abbildung: TSS Prozess für  $K = 10000$

# Optimierung

Außerdem verbessert sich damit die Lesbarkeit. Damit ist es möglich die Mutationspunkte und deren Auswirkungen genauer zu studieren:

# Optimierung

Außerdem verbessert sich damit die Lesbarkeit. Damit ist es möglich die Mutationspunkte und deren Auswirkungen genauer zu studieren:

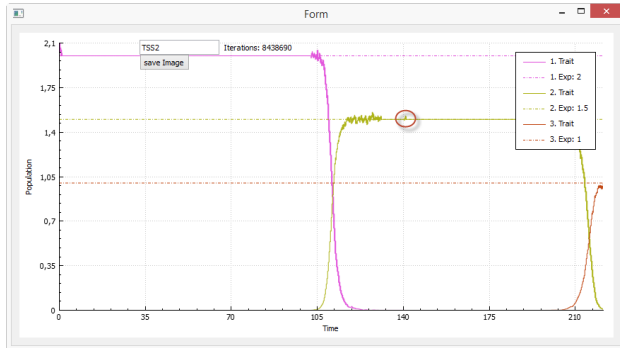


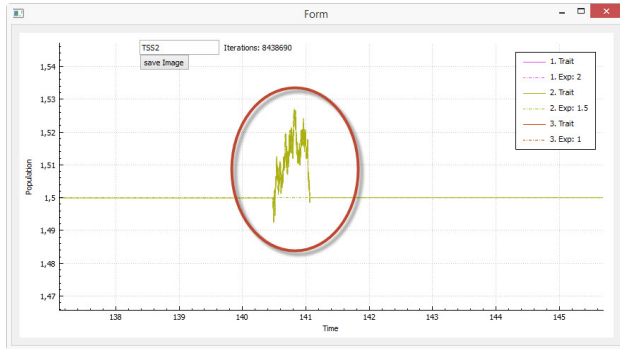
Abbildung: Invasionsversuch deutlich zu erkennen

# Optimierung

Näher betrachtete Auswirkungen:

## Optimierung

Näher betrachtete Auswirkungen:



**Abbildung:** Nahaufnahme eines Invasionsversuchs bei dominantem Merkmal

## Optimierung

Näher betrachtete Auswirkungen:

Ziel

Model

Normalisierung  
Equilibrium

Algorithmus

Simulation

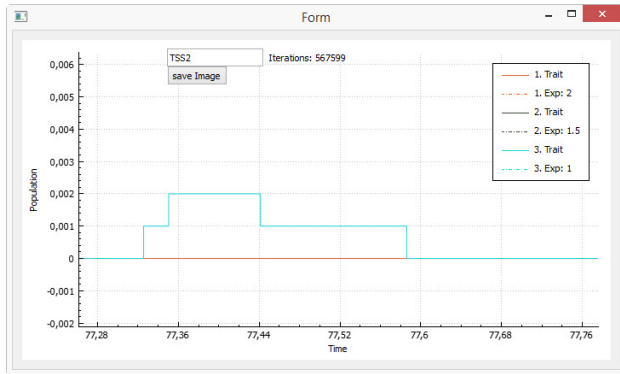
Aufgabenteilung  
und  
Flexibilität  
LayoutKorrektheit  
der Imple-  
mentationTSS -  
ProzesseFitness  
Interpolation

Abbildung: Nahaufnahme eines fehlgeschlagenen Invasionsversuchs

# Mutatinospunkte

Wie werden die Zeitpunkte für Mutationen bestimmt?



# Mutatinospunkte

Wie werden die Zeitpunkte für Mutationen bestimmt? Mit Raten!

- Geburtsraten der toten Merkmale

# Mutatinospunkte

Wie werden die Zeitpunkte für Mutationen bestimmt? Mit Raten!

- Geburtsraten der toten Merkmale

$$B(x) = \underbrace{0}_{\text{int. Geb.}} + \left( \underbrace{0}_{\text{Mut. von Totem}} + \underbrace{b(y) \cdot n_t(y)}_{\text{Mut. von Dom.}} \right) \cdot \frac{\mu}{2}$$

Mit dieser Mutationsrate wird eine neue Uhr gestellt die klingelt sobald sich eine Mutation ereignet.