# 数理生物学演習

第６回　突然変異固定までの待ち時間

## 第5回：突然変異固定までの待ち時間

本日の目標

- ハーディ-ワインベルグ平衡の導出
- ライト-フィッシャー モデルの解析
- 擬似乱数

# ハーディー-ワインベルグ モデル

世代を越えて遺伝子型頻度が維持されるケース

**仮定**

二倍体の有性生殖する生物を考える.
次のような性質を持つと仮定する.

- ランダム交配する
- 世代は重ならない
- 突然変異は起こらない

この生物の十分大きな集団において
中立な対立遺伝子Aとaに注目する.

ある世代における遺伝子型AA, Aa, aa それぞれの頻度は, 前の世代の配偶子中のAとaの頻度 $p$と$q$（ただし$p+q=1$）を用いて,

$$\mathrm{AA}:\mathrm{Aa}:\mathrm{aa}=p^2:2pq:q^2$$

となる.

この遺伝子型頻度は世代を越えて維持され, ハーディ-ワインベルグ平衡と呼ばれる.

**次世代の遺伝子型頻度を計算しよう**
**→ 遺伝子型頻度が維持されているか確認する**

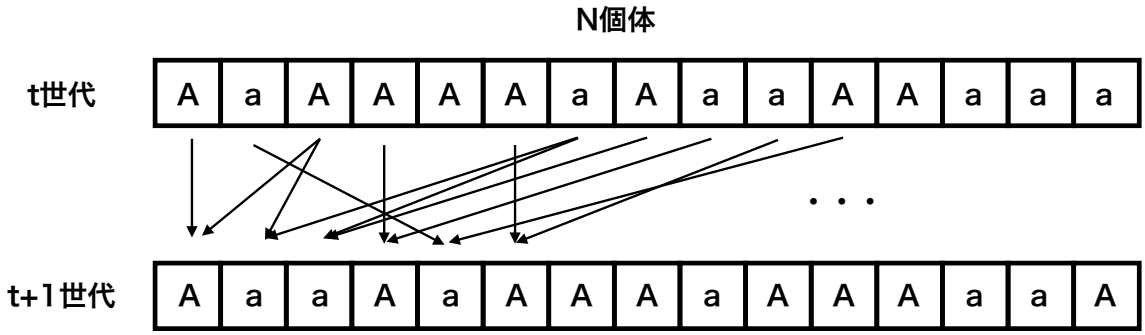
# ライト-フィッシャー モデル

遺伝的浮動を考えてみる

半数体の生物**N個体からなる集団**について, ある中立な対立遺伝子A, aに注目する.
次のような性質を持つとする.

- ランダム交配する
- 世代は重ならない
- （追加での）突然変異は起こらない

**もう少し詳しく知りたい人はHartl & Clark, Principles of Population Genetics 4th ed.の第3章がおすすめ**




集団サイズはN個体で一定
親個体をランダムに選び, どちらの遺伝子を引き継ぐかはランダムに決まる

# 実際にプログラムを組んでみよう！

# 擬似乱数

- ある確率分布に従うランダムな数値の系列（乱数列）を生成したいが，“真に”ランダムな数値を得ることは難しい
- **決定論的なアルゴリズム**によって本当の乱数列と（特定の目的上）区別がつかない数値（擬似乱数）列を生成し，これで代替することが一般的
- Pythonもrandomモジュールではメルセンヌ・ツイスタと呼ばれる疑似乱数生成器が使われている

- random.randrange(終了)
- random.randrange(開始，終了)
- random.randrange(開始，終了，ステップ)
  それぞれ，range(終了)，range(開始，終了)，range(開始，終了，ステップ)の要素からランダムに一つ要素を返す．
  例）random.randrange(100)：0〜99の中からランダムに返す．

```
# 6-1. 擬似乱数列の生成
import random

for i in range(100):
    r = random.randrange(1,1001)
    print(r)
```

```
出力
291
602
997
. . .
891
```

**毎回異なる結果が表示される**

**注意：疑似乱数関係のモジュール（randomモジュール）をimportする**

# 擬似乱数の種（シード）

- シードを指定すると擬似乱数列は一意に決まる.
- 同じ疑似乱数列を利用できると便利なケースがある（テスト，シミュレーションの再現性など）.
- シードをどのように設定するかは状況によるが，本演習では通常はNoneでの初期化を用いる．それ以外のものを特別に設定する場合は指示する.

- random.seed(シード)
  乱数生成器の初期化．乱数の種（シード）を設定する

```
# 6-2. シード
import random

random.seed(1)
for i in range(100):
    r = random.randrange(1,1001)
    print(r)
```

```
出力
138
583
868
 . . .
311
```

**乱数の種（シード）が同じなので毎回同じ結果が表示される**

# その他の乱数生成：シーケンス

シーケンス（リストなど）に対してランダムな処理（シャッフルやサンプリング）をおこなう

- random.choice(シーケンス)
  シーケンスからランダムに要素を返す
- random.choises(シーケンス，weights=重み，k=要素数)
  シーケンスから相対的な重みweightに基づき，（重複を許し）ランダムにk個の要素からなるリストを返す
- random.sample(シーケンス，k)
  シーケンスから重複のないk個の要素からなるリストを返す

```
# 6-3. シーケンス操作
import random

aList = [23, 22, 32, 12, 31, 30, 3, 35, 26, 36]
# choice
print("# random.choice")
for i in range(50):
    r = random.choice(aList)
    print(r)
print("")
```

```
# choices
weights = [0,1,1,1,1,1,1,1,1,10]
print("# random.choices")
for i in range(50):
    r = random.choices(aList, weights=weights, k
= 5)
    print(r)
print("")

# sample
print("# random.sample")
for i in range(50):
    r = random.sample(aList, k = 5)
    print(r)

# シャッフル
print("# random.sample シャッフル”)
for i in range(50):
    r = random.sample(aList, k = len(aList))
    print(r)
```

重みを指定しない場合は，すべて同じ重み

# その他の乱数生成：連続確率分布

特定の確率分布に従う擬似乱数列を生成する

- random.random()
  [0,1)の範囲の浮動小数点数（float型）の値をランダムに返す（一様分布）．
- random.uniform(a, b)
  一様分布．a≦x≦b（a>bならb≦x≦a）の範囲のランダムな浮動小数点数xを返す．
- random.gauss(mu, sigma)
  平均mu，標準偏差sigmaの正規分布に従う浮動小数点数を返す．

```
# 6-4. 連続確率分布
# 一様分布
# random
rListUni1 = []
for i in range(50):
    r = random.random()
    rListUni1.append(r)
print(rListUni1, "\n")

# uniform
rListUni2 = []
for i in range(50):
    r = random.uniform(5, 10)
    rListUni2.append(r)
print(rListUni2, "\n")

# 正規分布
rListGauss = []
for i in range(50):
    r = random.gauss(0, 1)
    rListGauss.append(r)
print(rListGauss, "\n")
```

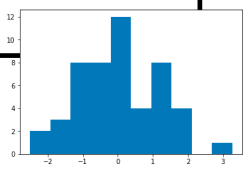
5以上10以下の一様乱数

平均0，標準偏差1の正規分布

- matplotlib.pyplot.hist(シーケンス)
  シーケンスのヒストグラムをプロット．

```
# 6-5. 分布のプロット
import matplotlib.pyplot as plt

plt.hist(rListGauss)
```



**他の分布やパラメータを変えていろいろプロットしてみよう！**

# ループからの脱出：break文

- ループから強制的に抜け出したいときがある
- forの範囲や条件分岐をうまく設定できれば良いが，
  細かく継続判定を設定し分割して判定するよう処理したい場合などに用いる

- break文
  一番内側のループを終了し，
  そのループの次の文へ処理が進む

```
# 6-5. ループの中断
for i in range(100):
    print(i)
    if i == 10:
        break
print("ループ終了")
```

**ループから脱出する**
**(i>10でforループは実行されない)**

```
出力
0
1
2
3
4
5
6
7
8
9
10
ループ終了
```

**forなどのループで利用できる**

```
# 6-6. ネストされたループの中断
for i in range(20):
    for j in range(20):
        print(i,", ",j)
        if j == 5:
            break
```

**breakした一番内側のループが終了する（その外側は通常どおり）**

```
出力
0 ,  0
0 ,  1
0 ,  2
0 ,  3
0 ,  4
0 ,  5
1 ,  0
1 ,  1
1 ,  2
1 ,  3
1 ,  4
1 ,  5
2 ,  0
2 ,  1
…
19 ,  4
19 ,  5
```

# サイコロの目の総和が100を超えるまでの待ち時間

```
# 6-7. サイコロの目の総和が100を超えるまでの待ち時間
import random

x = 0
for i in range(100):
    r = random.randrange(1,7)
    x = x + r
    print(str(i+1)+"回目：",x)
    if x >= 100:
        print(str(i+1)+"回目で100を超えた．")
        break
```

x：サイコロの目の総和

xが100以上ならばステップ数を出力してループを脱出

**サイコロ（6面ダイス）のモデル**

```
random.randrange(1,7)
```

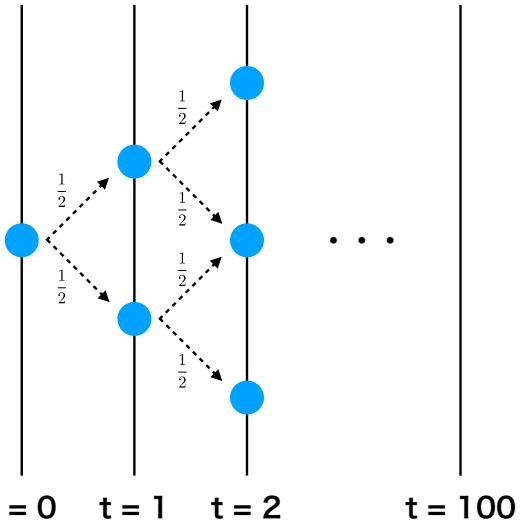
1以上6以下の整数をランダムに（一様分布に従って）返す

```
出力
1回目： 4
2回目： 9
3回目： 12
…
25回目： 96
26回目： 101
26回目で100を超えた．
```

**（〜回目の部分は）実行するたびに変わる**

# ランダムウォーク

以下のような1次元の単純ランダムウォークをシミュレーションしてみよう．



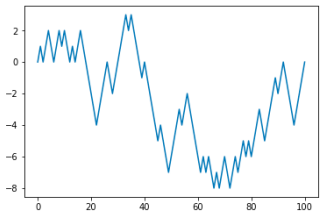

解釈例：

- コインをなげて表が出れば+1点，裏が出れば-1点を得るゲームの点数の推移
- 右と左にそれぞれ1/2の確率で移動する生物の移動軌跡

```
# 6-8a. ランダムウォーク choice
x = 0
xList = [x]
for i in range(100):
    r = random.choice([-1,1])
    x = x + r
    xList.append(x)
```

```
# 6-8b. ランダムウォーク randrange
x = 0
xList = [x]
for i in range(100):
    r = random.randrange(-1,2,2)
    x = x + r
    xList.append(x)
```

**結果をプロットしてみよう**



# ライト-フィッシャー モデル

```
# 6-9. ライト-フィッシャー モデル
numPop = 100 # 個体数
genFin = 100 # 最終世代

a = [] # 集団の注目している対立遺伝子を記録するリスト
aa = [] # 次世代集団の注目している対立遺伝子を記録するリスト

for i in range(numPop):
    if i % 2 == 0:
        a.append(0)
    else:
        a.append(1)
print(a)

# aaの初期化
for i in range(len(a)):
    aa.append(0)

for t in range(genFin):
    for i in range(numPop):
        p1 = random.randrange(numPop)
        p2 = random.randrange(numPop)
        r = random.choice([a[p1],a[p2]])
        aa[i] = r
    for i in range(numPop):
        a[i] = aa[i]
print(a)
```

初期値として半分の個体が『0』もう半分の個体が『1』を持つとする

各個体の持つ対立遺伝子を出力

各個体の持つ対立遺伝子を出力

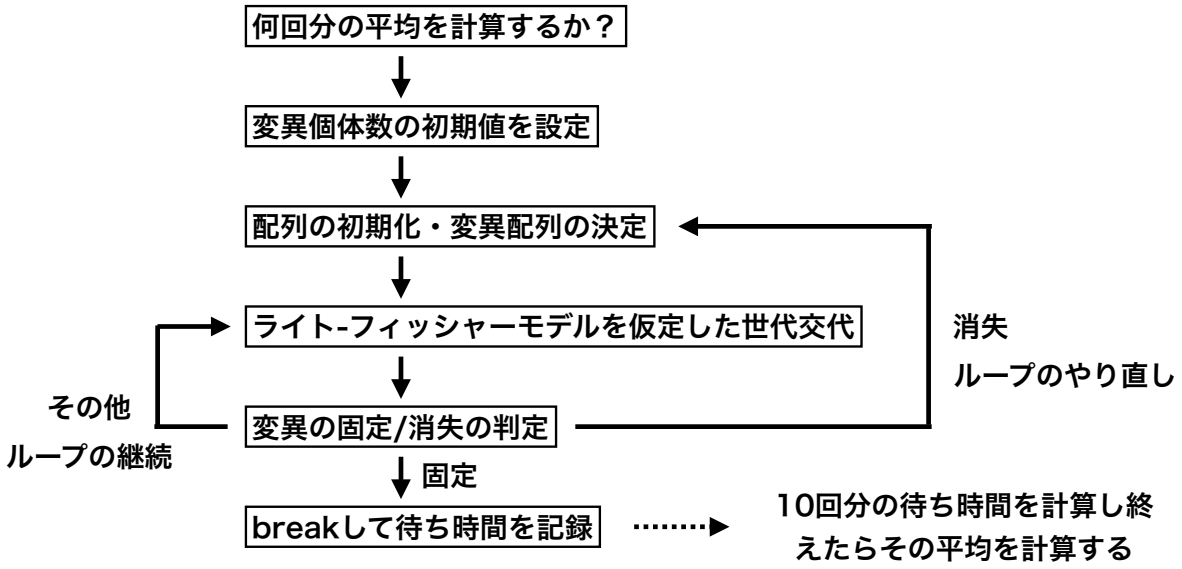
p1番目とp2番目の個体が親として選ばれる

どちらの親から遺伝子を引き継ぐか，確率1/2でランダムに決まる

# 突然変異固定までの待ち時間

ライト-フィッシャー モデルを仮定し，突然変異固定までの待ち時間を計算する

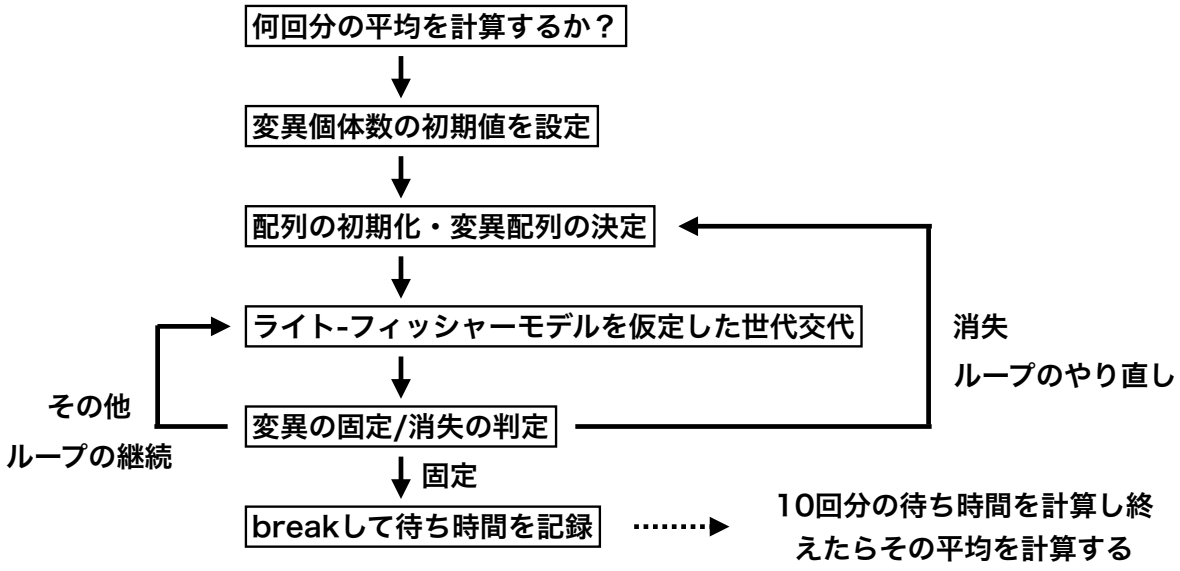
```
# 6-10. 突然変異固定までの平均待ち時間を10回分について計算してみよう
```



# 突然変異固定までの待ち時間

ライト-フィッシャー モデルを仮定し，突然変異固定までの待ち時間を計算する

```
# 6-10．突然変異固定までの平均待ち時間を10回分について計算してみよう
```




# 本日の課題

1. 6-7を改良し，サイコロの目の総和が100を超えるまでの平均待ち時間を試行数10回，100回，1000回，10000回についてそれぞれ計算せよ.
2. ランダムウォークの結果を5つ重ねてプロットせよ
3. ライト-フィッシャーモデルを仮定し世代交代を繰り返し，集団に突然変異が固定する場合について，その100回分の平均待ち時間Tを求めよ

**ハード** 4. N個体の集団内に占める突然変異対立遺伝子を持つ個体の数をk，その頻度をp（=k/N）とする．N=100，N=200の場合について，それぞれkを1〜NまでN/10刻みで変化させ，突然変異の初期頻度pに対する平均待ち時間Tを10個ずつプロットせよ.

**ハード** 5. 半数体生物に対して突然変異固定までの平均待ち時間Tの解析解が

$$T(p) = -\frac{1}{p}\{2N(1-p)\log_e(1-p)\}$$

で与えられるとき，このグラフをN=100，N=200について描き，同じグラフ上で上記のプロットと比較し，考察せよ.

6. 質問，意見，要望等をどうぞ.

**課題をPDFファイルにまとめて，Moodleにて提出すること**

# 次回予告
# 第７回：理論形態モデル
# ６月３日

## 復習推奨

- 指数増殖モデル
- 回転行列

# 宣伝
# 数理生物学 第７回
# かたちの数理モデル（２）
# 5月２９日（水）
# 内容

- 画像解析
- 形態測定学
  - 標識点ベース形態測定学
  - 輪郭ベース形態測定学