先端生命科学実験 I
バイオインフォマティクス

遺伝的アルゴリズムによるタンパク質立体構造の重ね合わせ

藤　博幸

この実験の目的

(1) プログラミング言語Rの使い方を学ぶ。

(2) 進化の分子機構と遺伝的アルゴリズムを理解する。

(3) 遺伝的アルゴリズムを具体的な生物の問題であるタンパク質立体構造の重ね合わせに応用する。（世代数、突然変異率の影響など）

OUTLINE

1. 遺伝的アルゴリズム
2. タンパク質の立体構造
3. Rの復習
4. 相同性の説明
5. 遺伝的アルゴリズムによる相同タンパク質の立体構造の重ね合わせ
6. レポートの構成

1. 遺伝的アルゴリズム

生物システムの情報科学への応用

- ニューラルネットワーク
- 遺伝的アルゴリズム
- 進化プログラミング
- 人工免疫システム
- 粒子群最適化
- 蟻コロニー最適化
- 人工生命

遺伝的アルゴリズム

菊川怜の卒論テーマ

「遺伝的アルゴリズムを適用したコンクリートの要求性能型の調合設計に関する研究」

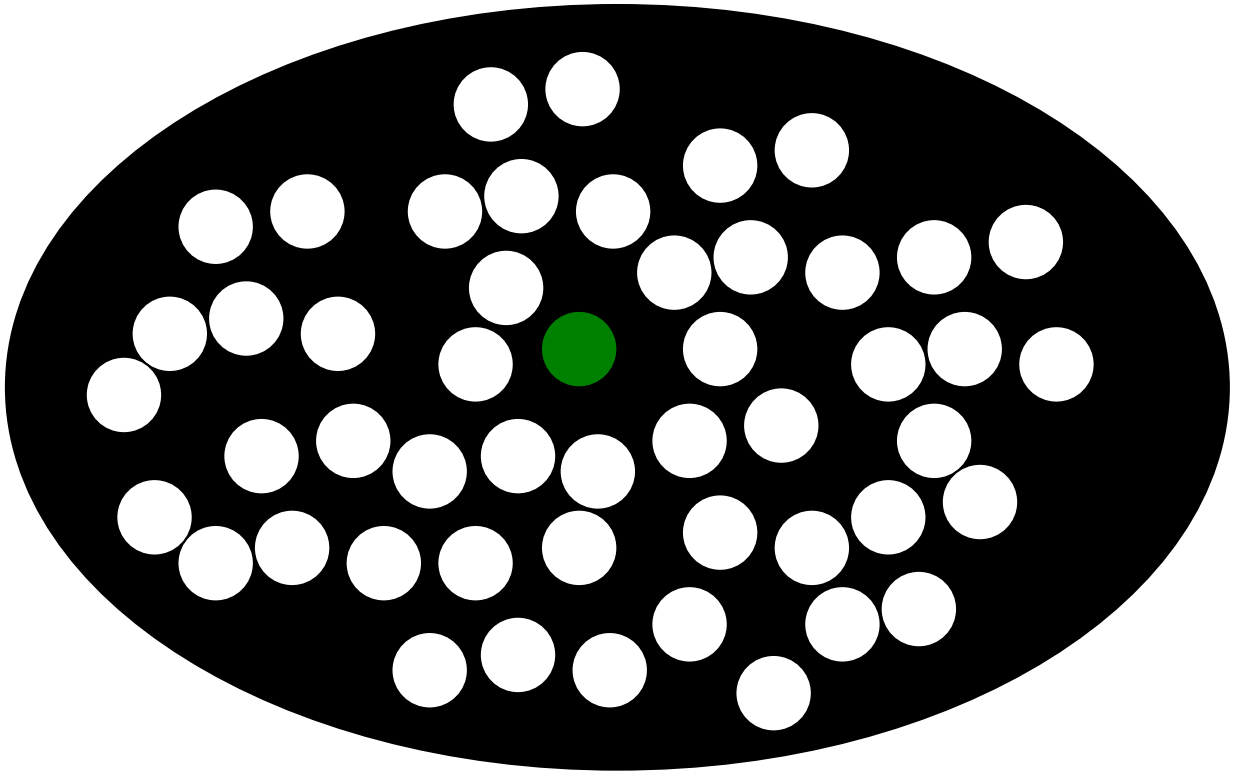
遺伝的アルゴリズム

Genetic Algorithm (GA)

1975年　John H. Hollandにより提案

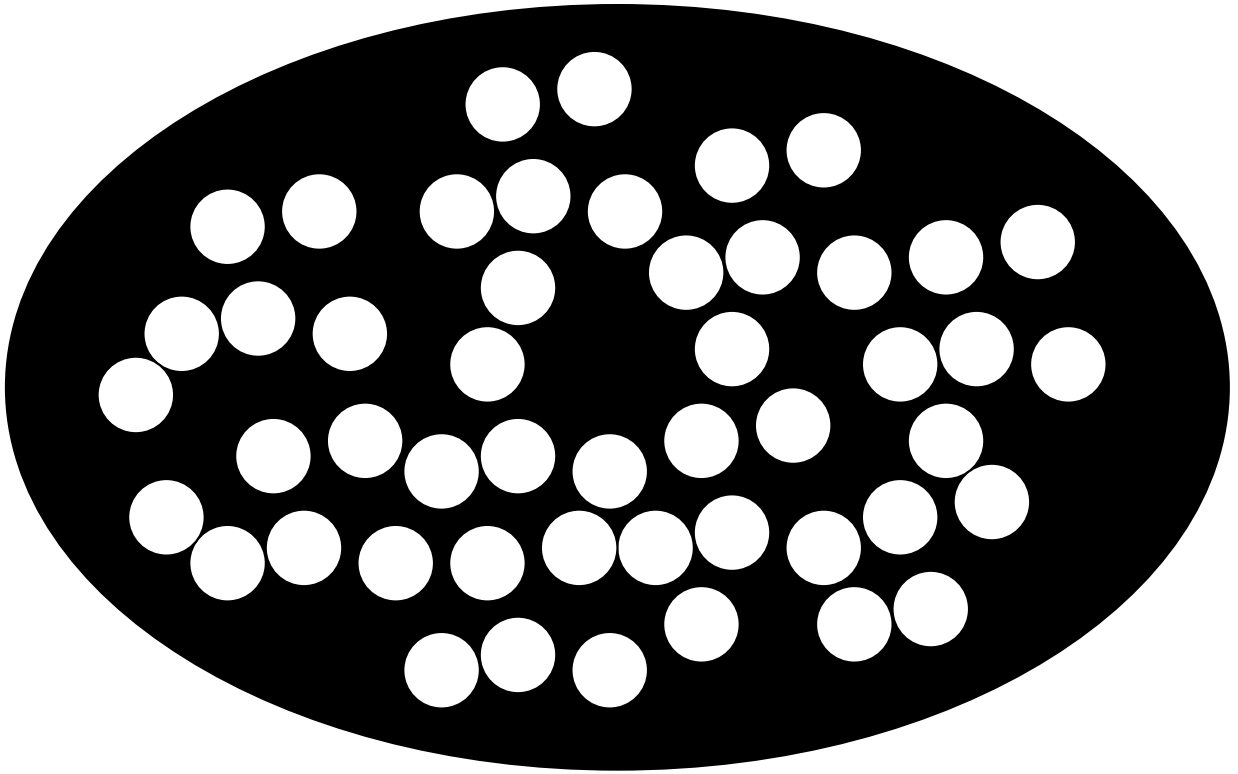
生物の遺伝の仕組みを模倣した最適化の手法
たとえば
ナップザック問題、ハミルトン閉路問題、
巡回セールスマン問題、施設配置問題
などの最適化問題に適用できる

突然変異(mutation)と置換（substitution)



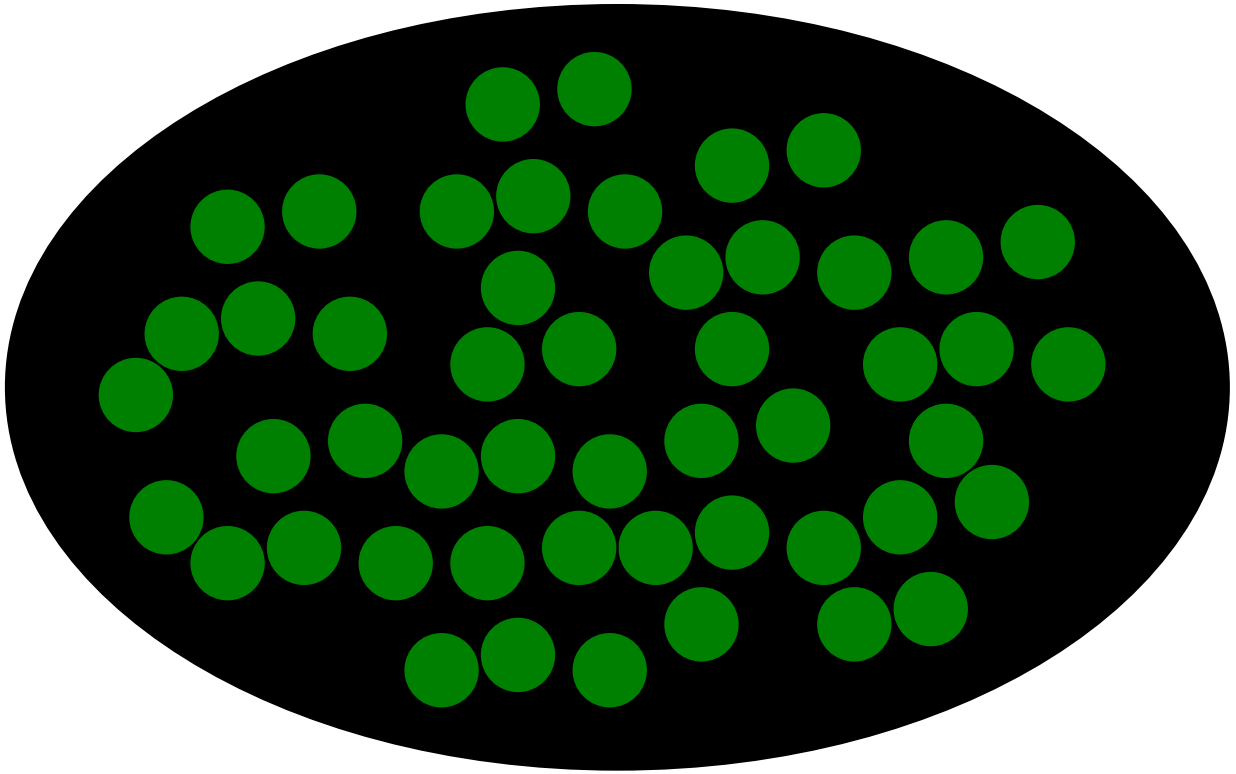
突然変異は集団中の個体に生じる
※　進化に寄与するのは体細胞ではなく生殖系列
　　の細胞に生じる突然変異

突然変異(mutation)と置換（substitution)



有害な突然変異は、集団から除去される
（負の選択 or 純化淘汰）

突然変異(mutation)と置換（substitution)



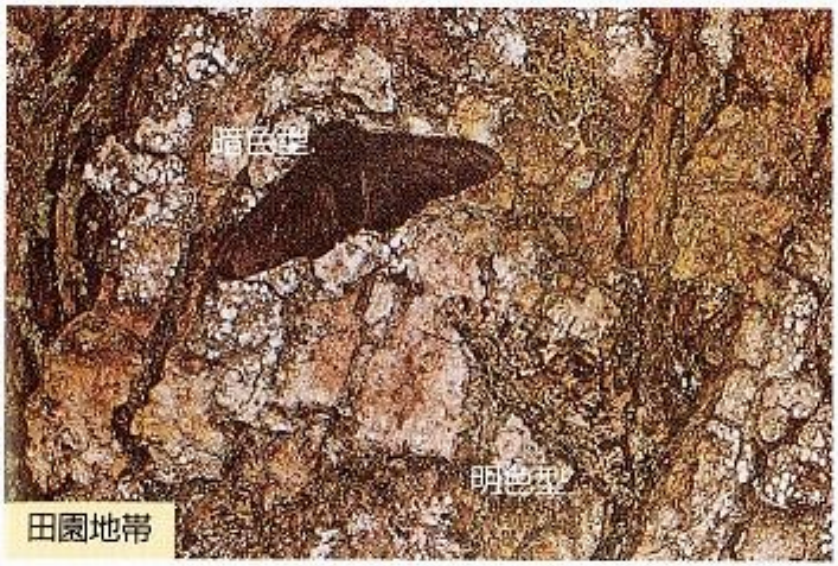
有利な突然変異は急速に集団中に広まり
集団全体がその突然変異遺伝子で置き換
えられる （正の選択）

工業暗化
(industrial melanism)

もともと淡色型と暗色型の体色を持つオオモリエダシャクが存在

19世紀後半から、ヨーロッパの工業都市が発展するにつれて、
その付近に生息するガ（蛾）に暗色の個体が増加した

田園都市では白っぽい地衣類が木にはえていて淡色型の方が
保護色となり鳥の補色を免れて生き残るのに対して、工業化に伴い
地衣類が枯れ、煤煙で木が黒くすすけてしまったため暗色型の
ほうが保護色となるため



樹皮は白っぽい地衣植物でおおわれており，
明るい色をしている。

大気汚染のために地衣植物がなくなり，
樹皮は暗い色をしている。

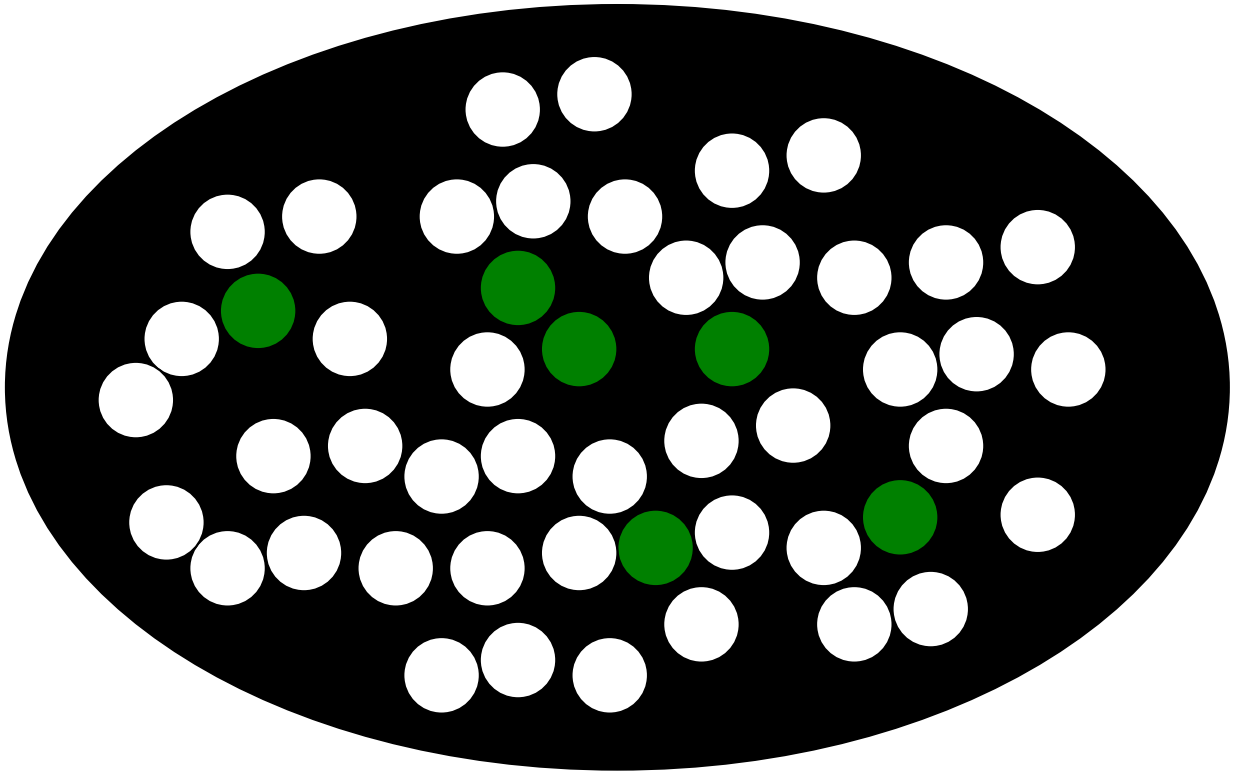
補足

大進化と小進化

工業暗化では、種内での形質の変化は起きているが
“種分化”はおきていない。このようなレベルの進化を小進化とよぶ

これに対し、新しい種あるいは種より高次の分類群の形成、また絶滅
などのレベルの進化を大進化とよぶ

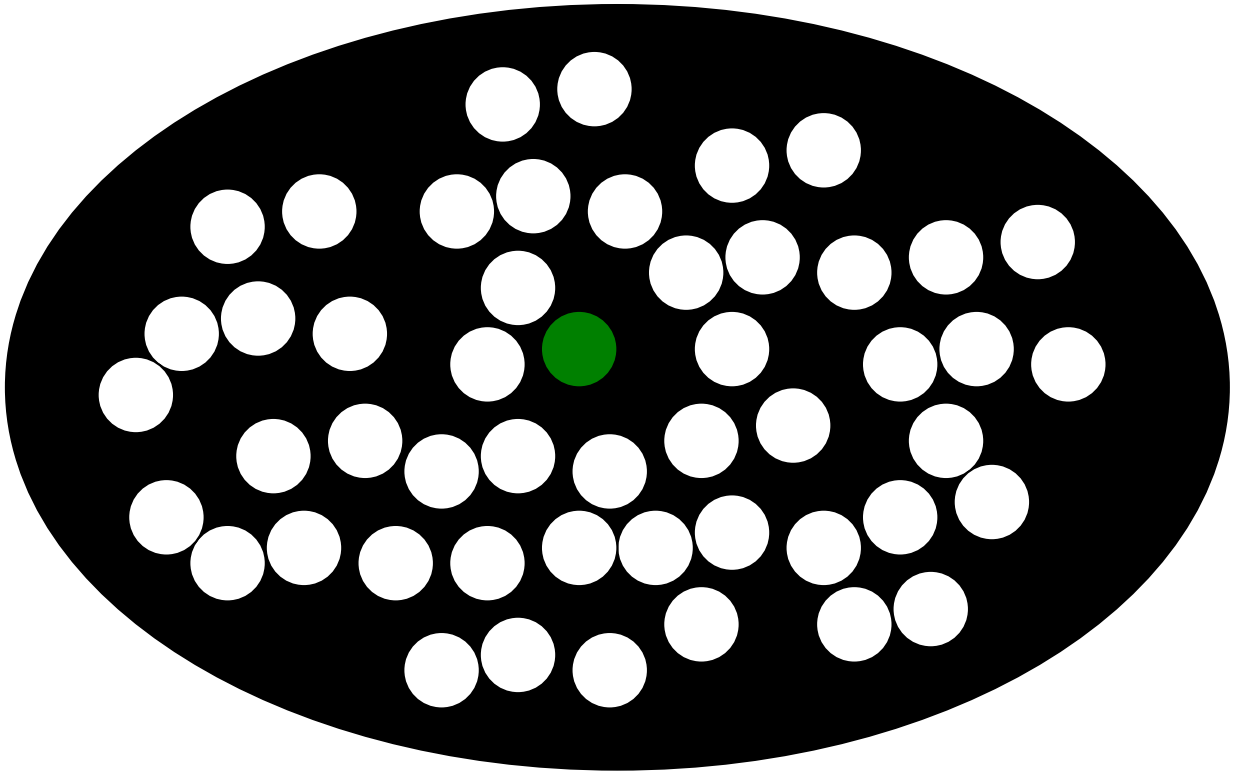
突然変異(mutation)と置換（substitution)



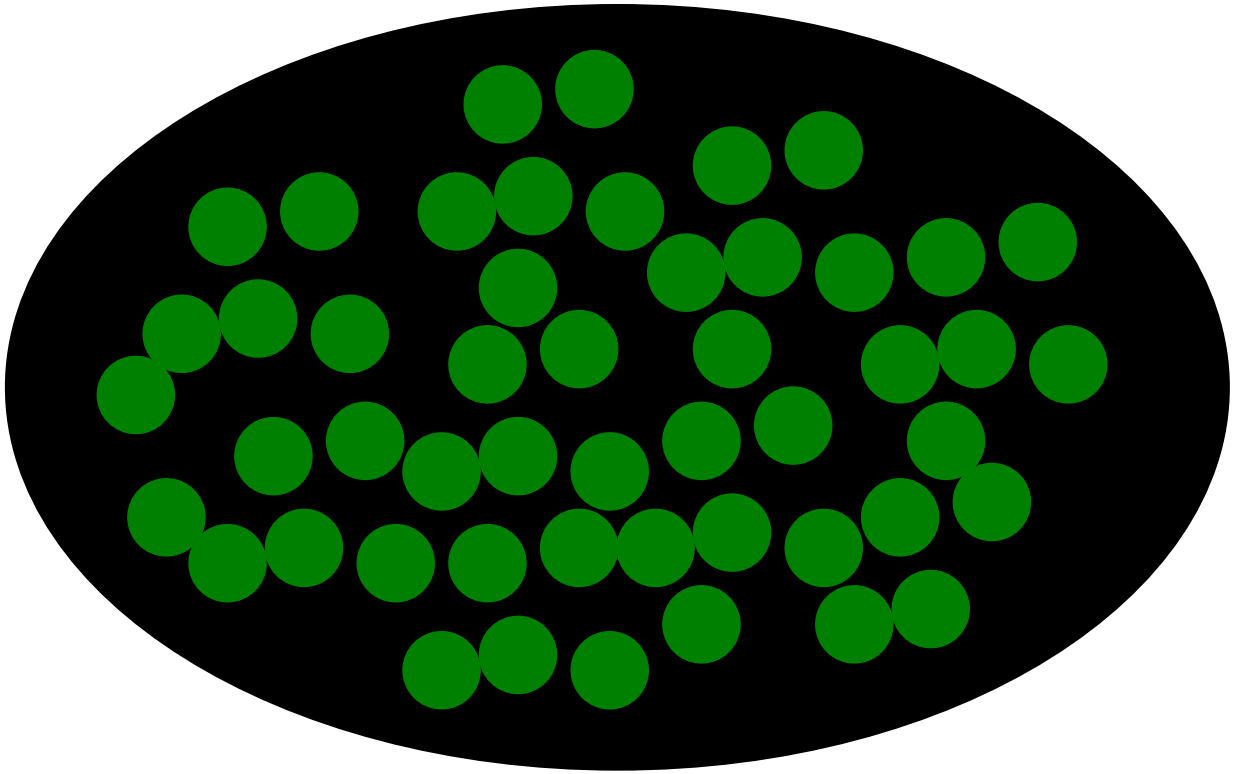
中立な突然変異の頻度はランダムに浮動し
確率的に集団中に固定 （選択的に中立）

固定にいたる途中の過程　　多型(polymorphism)

突然変異(mutation)と置換（substitution)



突然変異(mutation)と置換（substitution)



個体に生じた突然変異が集団全体に広まること：**固定**

突然変異が固定されること：**置換**

LETTER van'tHof *et al*. (2016) *Nature* **534**, 102-105

doi:10.1038/nature17951

The industrial melanism mutation in British peppered moths is a transposable element

Arjen E. van't Hof[1]*, Pascal Campagne[1]*, Daniel J. Rigden[1], Carl J. Yung[1], Jessica Lingley[1], Michael A. Quail[2], Neil Hall[1], Alistair C. Darby[1] & Ilik J. Saccheri[1]

Discovering the mutational events that fuel adaptation to environmental change remains an important challenge for evolutionary biology. The classroom example of a visible evolutionary response is industrial melanism in the peppered moth (*Biston betularia*): the replacement, during the Industrial Revolution, of the common pale *typica* form by a previously unknown black (*carbonaria*) form, driven by the interaction between bird predation and coal pollution[1]. The *carbonaria* locus has been coarsely localized to a 200-kilobase region, but the specific identity and nature of the sequence difference controlling the *carbonaria–typica* polymorphism, and the gene it influences, are unknown[2]. Here we show that the mutation event giving rise to industrial melanism in Britain was the insertion of a large,

involved in wing pattern development or melanization. By extending the association mapping approach to a larger population sample and more closely spaced genetic markers (see Methods), we narrowed the *carbonaria* candidate region to about 100 kb (Fig. 1a). The candidate region resides entirely within the span of one gene — the orthologue of *Drosophila cortex* (*cort*), the only known function of which is as a cell-cycle regulator during meiosis[11]. In *B. betularia*, *cortex* consists of eight non-first exons, multiple alternative first exons (of which only two, 1A and 1B, are strongly expressed in developing wing discs), and a very large first intron (Fig. 1b).

The rapid spread of *carbonaria* gave rise to strong linkage disequilibrium[2], such that many sequence variants are associated with the *carbonaria* phenotype. This poses a challenge for isolating the specific

オオシモフリエダシャクの17番染色体　cortex遺伝子

ショウジョウバエでは減数分裂時の細胞周期の制御に関与

黒い蛾のcortex 遺伝子第一イントロンにトランスポゾン

羽化時に発現が上昇するが、トランスポゾン挿入によるさらに発現が上昇

まだ、この遺伝子が工業暗化をもたらすことの機構までは解明されていない

適応度 （fitness)

ある個体の子供の内の、繁殖年齢まで達したものの数

遺伝的アルゴリズム

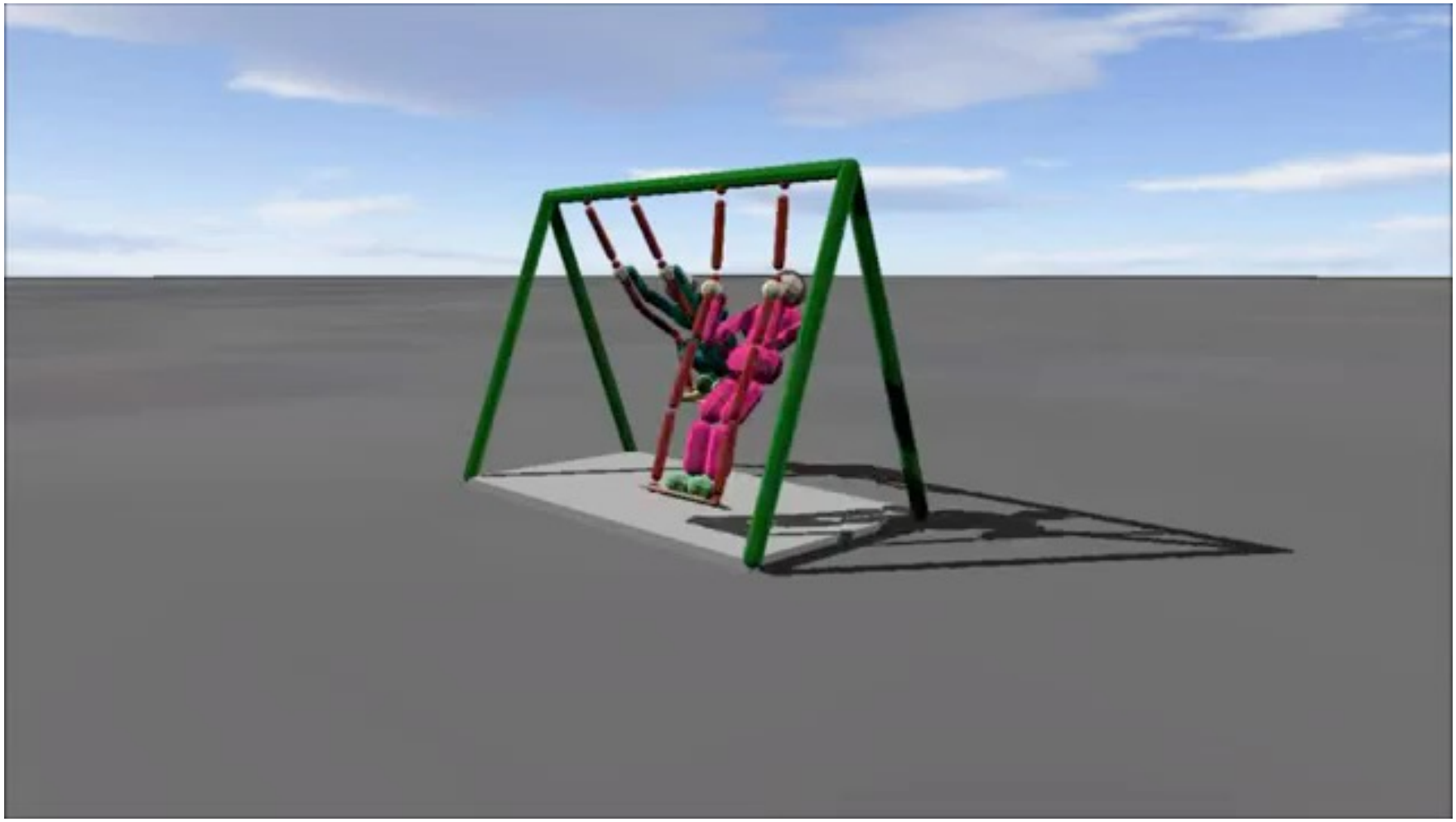
Genetic Algorithm (GA)
1975年　John H. Hollandにより提案

生物の遺伝の仕組みを模倣した最適化の手法
たとえば
ナップザック問題、ハミルトン閉路問題、
巡回セールスマン問題、施設配置問題
などの最適化問題に適用できる

巡回セールスマン問題を例として遺伝的アルゴリズムを学ぶ

遺伝的アルゴリズムでブランコの漕ぎ方を学習させた。

http://www.youtube.com/watch?v=8vzTCC-jbwM#t=14



物理エンジン

物理エンジン:コンピュータ上の**仮想的な3次元空間において重力、摩擦力**などを自動計算して**物体の動きをシミュレーションするソフトウェア**

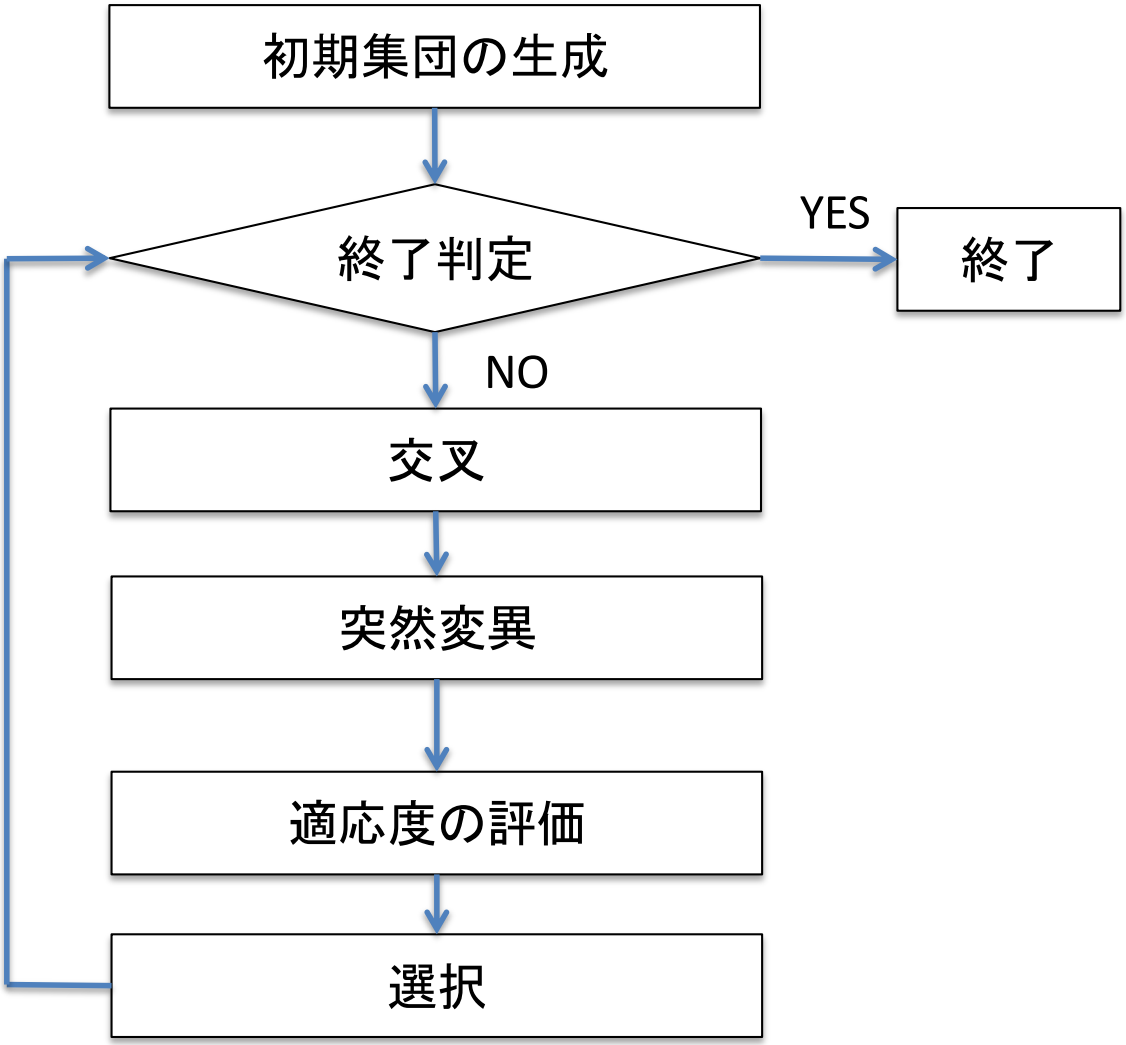
Chipmunk Physics
http://chipmunk-physics.net/

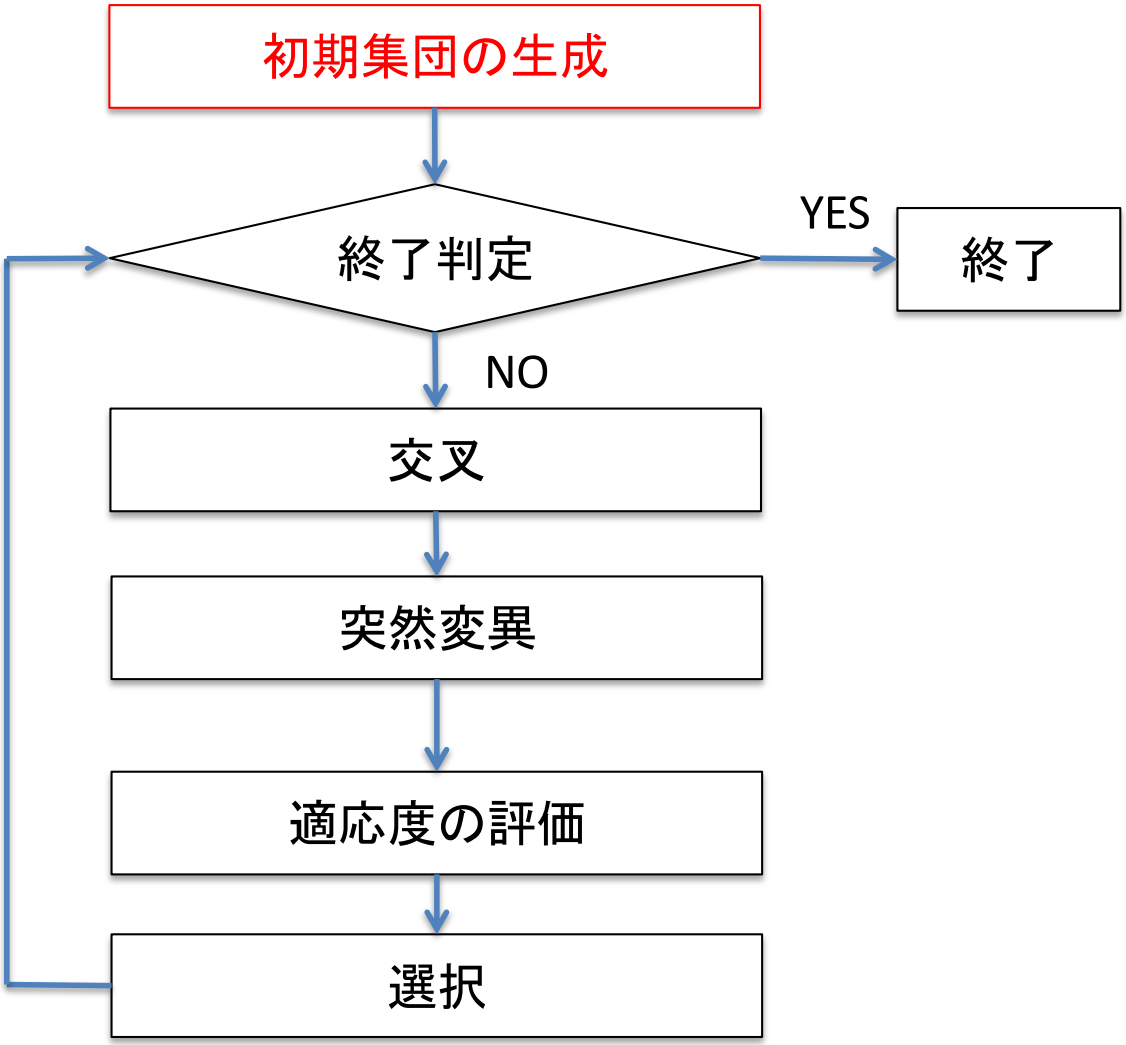
PhysX
http://developer.nvidia.com/object/physx_downloads.html

BULLET
http://code.google.com/p/bullet/downloads/list

Dynamo
http://home.iae.nl/users/starcat/dynamo/dynamo.zip

Springhead
http://springhead.info/wiki/index.php?plugin=attach&refer=Springhea...


初期集団の生成
終了判定
YES
終了
NO
交叉
突然変異
適応度の評価
選択


初期集団の生成
終了判定
YES
終了
NO
交叉
突然変異
適応度の評価
選択

初期集団の生成

集団サイズを3とする

３個体を生成し、ブランコの立った座ったを10でランダムに表現

個体1:１１１００ １… ０１０１１

個体2:０１００１ １… １００１０

個体3:００１１１ ０… １１０００

用語の説明

個体（ Individual ）:　染色体（後で説明）によって
特徴づけられた自律的な個

集団（ Population ）: 個体の集まり

集団サイズ（ Popolation Size ）: 集団内の個体数

用語の説明

- **遺伝子**（ Gene ）: 個体の形質を規定する基本構成要素

個体2の場合 ０１００１ １… １００１０の1, 0のそれぞれが遺伝子

- **染色体**（ Chromosome ）: 複数の遺伝子の集まり

個体2の場合、０１００１ １… １００１０の全体が染色体

- **遺伝子座**（ Locus ）: 染色体上の遺伝子の位置

- **対立遺伝子**（ Allele ）: 遺伝子がとりうる値

遺伝子座2は、0, 1の2つの値をとりうる。
ここの例では、個体1と2では1、個体3で0

個体1:１１１００ １… ０１０１１
個体2:０１００１ １… １００１０
個体3:００１１１ ０… １１０００

用語の説明

- **遺伝子型**（ Genotype ）: 染色体の内部表現

- **表現型**（ Phenotype ）: 染色体によって規定される形質の外部的表現. 巡回セールスマン問題の場合、巡路

	表現型	**遺伝子型**
個体1	ブランコの漕ぎ方1	１１１００ １… ０１０１１
個体2	ブランコの漕ぎ方2	０１００１ １… １００１０
個体3	ブランコの漕ぎ方 3	００１１１ ０… １１０００

- **コード化**（ Coding ）: 表現型から遺伝子型へ変換すること

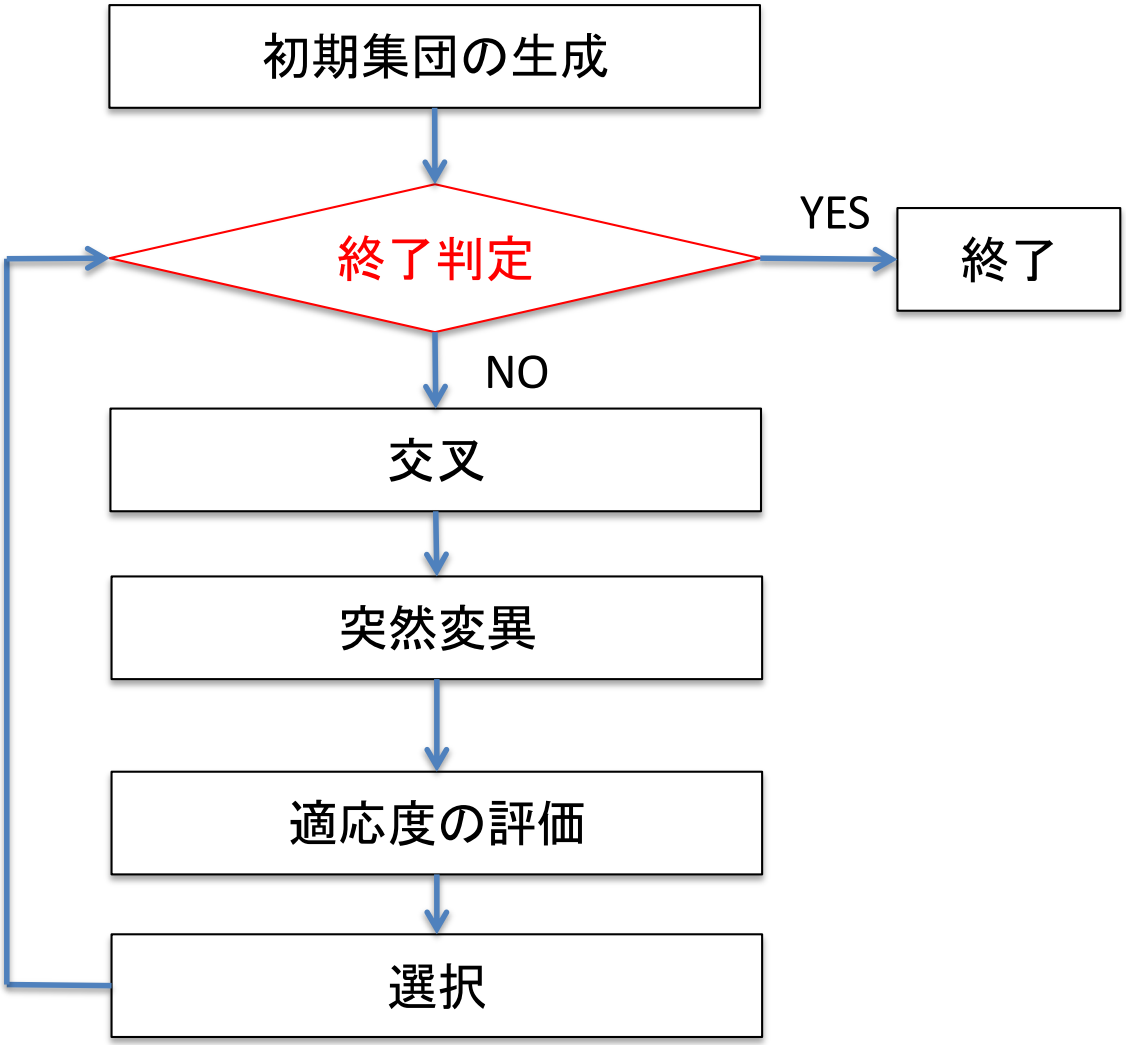
- **デコード化**（ Decoding ）: 遺伝子型から表現型変換すること

初期集団完成

個体1:１１１００ １… ０１０１１

個体2:０１００１ １… １００１０

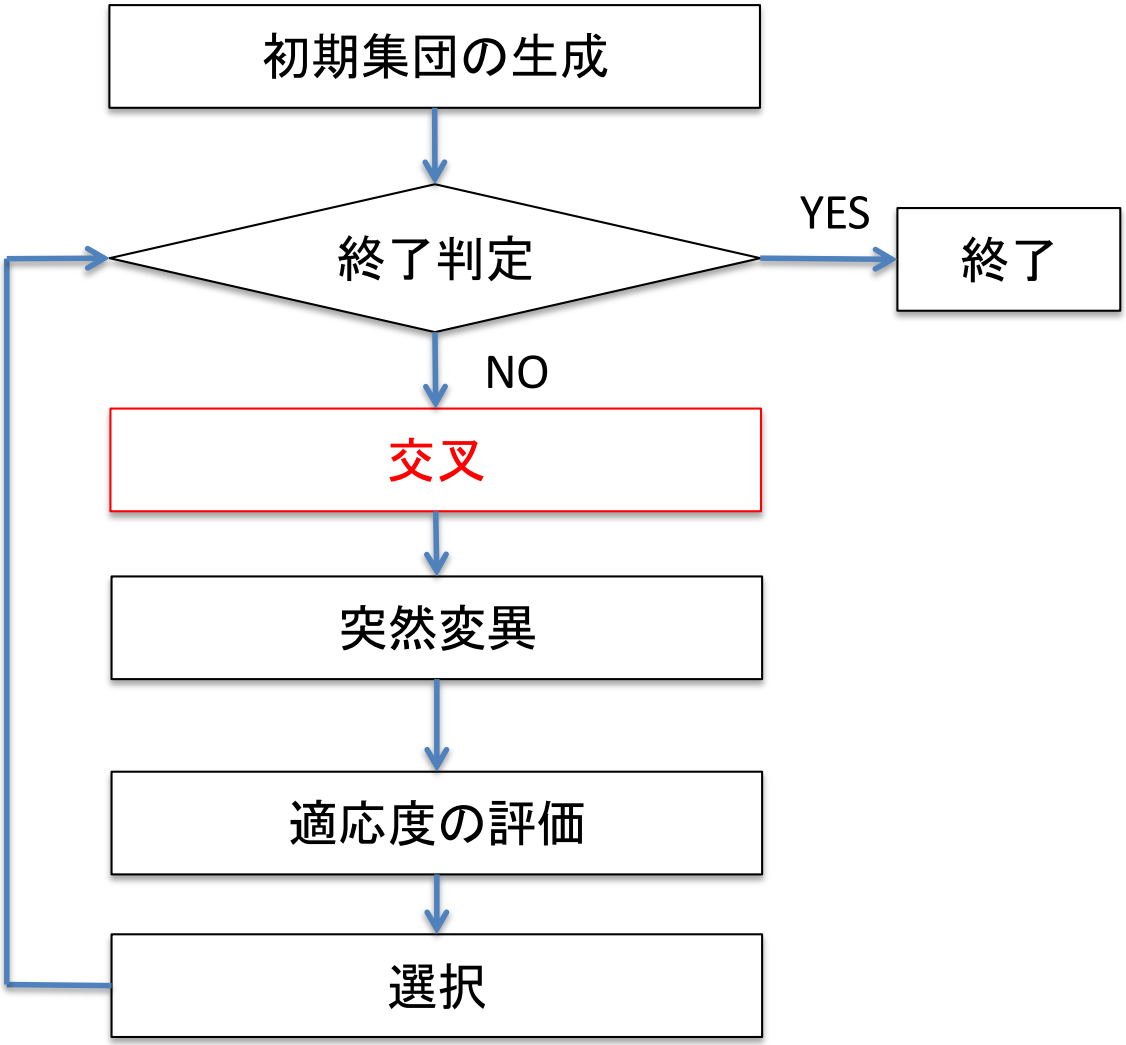
個体3:００１１１ ０… １１０００


初期集団の生成
終了判定
YES
終了
NO
交叉
突然変異
適応度の評価
選択

終了の判定

判定方法としていくつかのものが考えられる

- 集団中の最大の適応度が，ある閾値より大きくなった場合
- 集団全体の平均適応度が，ある閾値より大きくなった場合
- 集団の適応度の増加率がある閾値以下になる世代が一定期間続いた場合
- 世代交代の回数が規定の回数を超えた場合


初期集団の生成
終了判定
YES
終了
NO
交叉
突然変異
適応度の評価
選択

交叉

- *N*個の親集団からランダムに2*M*個の個体を選択
- ペアごとに組替え処理
- *N* + 2*M*個の個体が生成される

今、*M* = 1とし、乱数を利用して次の2個体が選択されたとする、

個体2:０１００１　１…　１００１０

個体3:００１１１　０…　１１０００

交叉の方法

- 1 点交叉法　染色体の切断箇所をランダムに 1 カ所指定し，その箇所（下の例における赤線の部分）で遺伝子を交叉させる．

  01001|101　　01001|110

  01100|110　　01100|101

- 多点交叉法　染色体の切断箇所をランダムに複数カ所（下の例では，2 カ所）指定し，それらの箇所で遺伝子を交叉させる．

  010|011|01　　010|001|01

  011|001|10　　011|011|10

- 一様交叉法　染色体と同じ長さのマスクと呼ばれるビット列を，0 に対しては確率 p，また，1 に対しては確率 $(1-p)$ で発生させ，マスクの値が 1 であるときは親A の遺伝子を子 C（親 B の遺伝子を子D）に，また，0 であるときは親 B の遺伝子を子 C（親 A の遺伝子を子 D）に継承させる．

  例えば，マスクが　00101011 となった場合は，以下のようになる．

  A　01001101　　C 01000101

  B　01100110　　D 01101110

- 個体の遺伝子型が連続値を表すときには，2 つの親の平均値を子の遺伝子とする，

  平均化交叉等の方法も使用される．平均化交叉では，2 つの親から 1 つの子供が生成される．

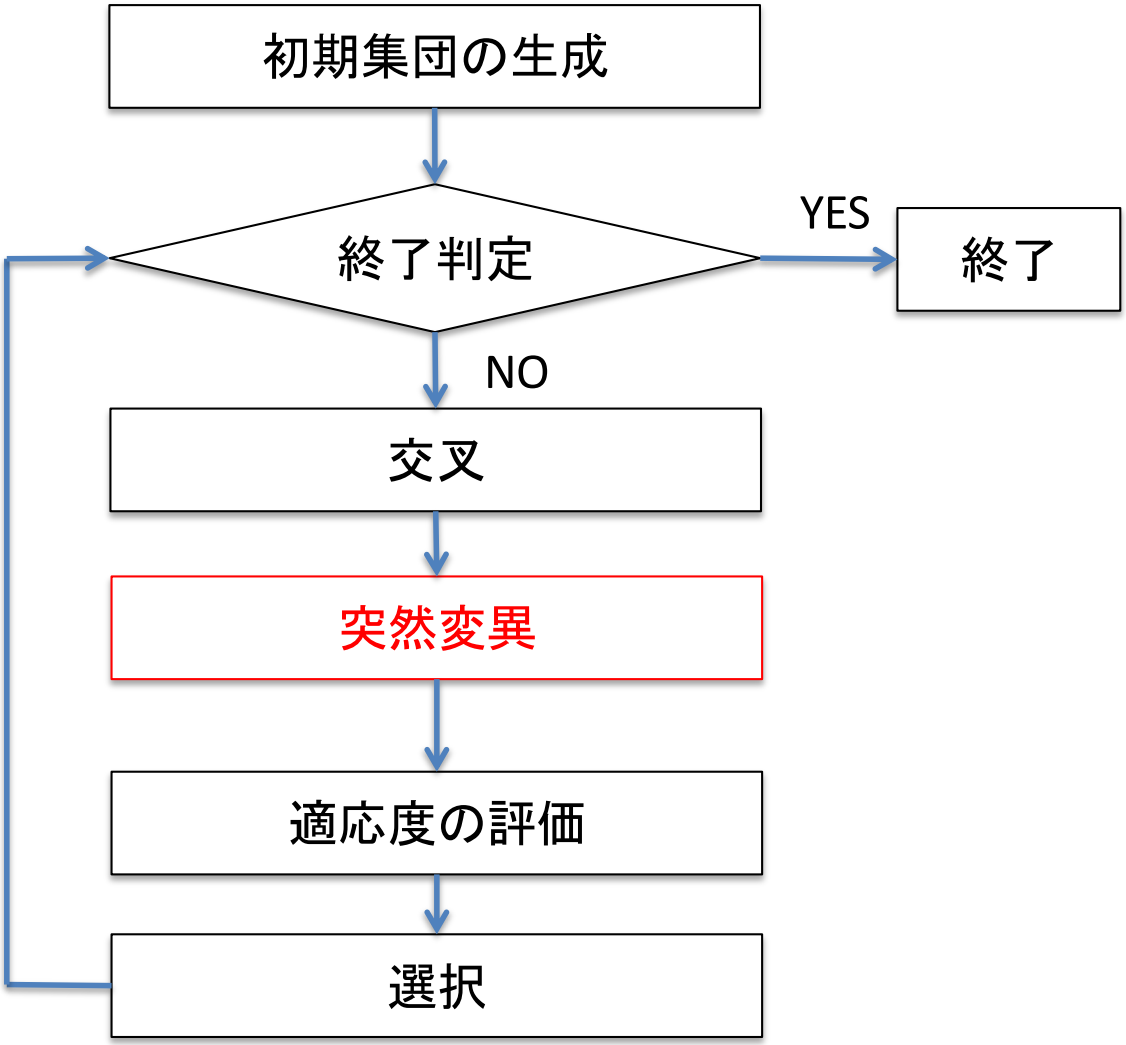
http://www.sist.ac.jp/~suganuma/kougi/other_lecture/SE/opt/GA/GA.htm#4.3より転載

一様交叉を選択された個体2と個体3に適用
　マスクが０１１０１１…　０１１０１とおく

　　０１１０１１…　０１１０１
個体2:００１０１０…　１１０１０
個体3:０１０１１１…　１００００

が得られる

子集団に交叉で得られた集団（サイズ2）を加える。


初期集団の生成
終了判定
YES
終了
NO
交叉
突然変異
適応度の評価
選択

突然変異

ある与えられた確率 P_m（突然変異率）で，突然変異を起こさせる。
突然変異率は通常小さな値（たとえば5%）に設定。

一般的な方法として、各遺伝子をランダムに対立遺伝子に置き換える。

今回の例の場合、親集団（サイズ3）の各遺伝子座（5個）について、0~1の一様乱数を発生させ、それが0.05より小さい場合、その遺伝子座の可能な対立遺伝子のいづれかに置換する。

- 個体1:1 1 1 0 0 1 … 0 1 0 1 1
- 個体2:0 1 0 0 1 1 … 1 0 0 1 0
- 個体3:0 0 1 1 1 0 … 1 1 0 0 0

ここで乱数が0.05より小さかったとする

個体3:0 0 1 1 1 0 … 1 1 0 0 0

遺伝子座2を、0から1に変更

0 1 1 1 1 0 … 1 1 0 0 0

という個体を生成し、子集団に加える。

0 1を入れ替える突然変異 = bit flip mutation

子集団

交叉によって生じた

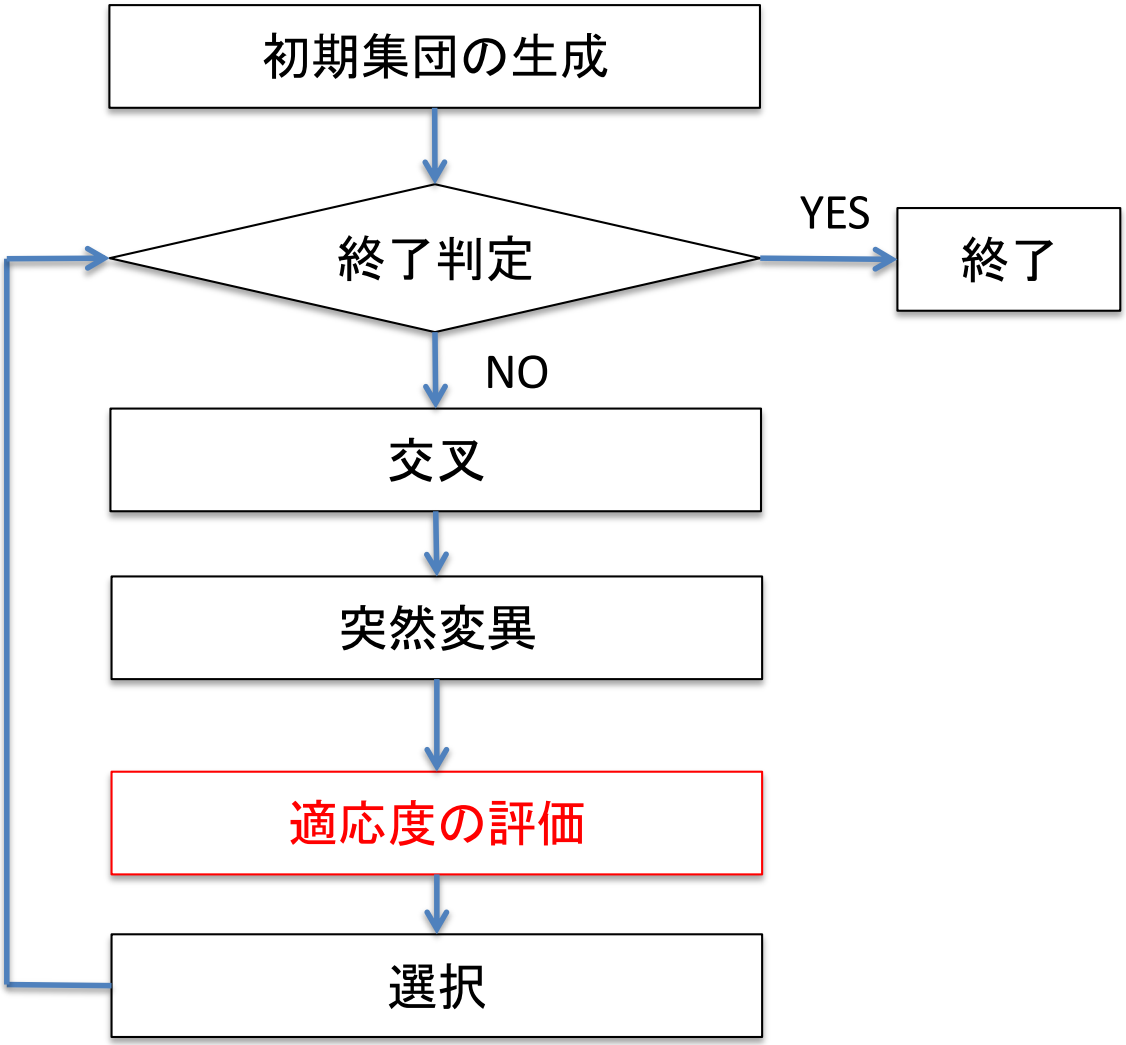
0 0 1 0 1 0 … 1 1 0 1 0

0 1 0 1 1 1 … 1 0 0 0 0

突然変異によって生じた

0 1 1 1 1 0 … 1 1 0 0 0

の3個体が含まれている。


初期集団の生成
終了判定
YES
終了
NO
交叉
突然変異
適応度の評価
選択

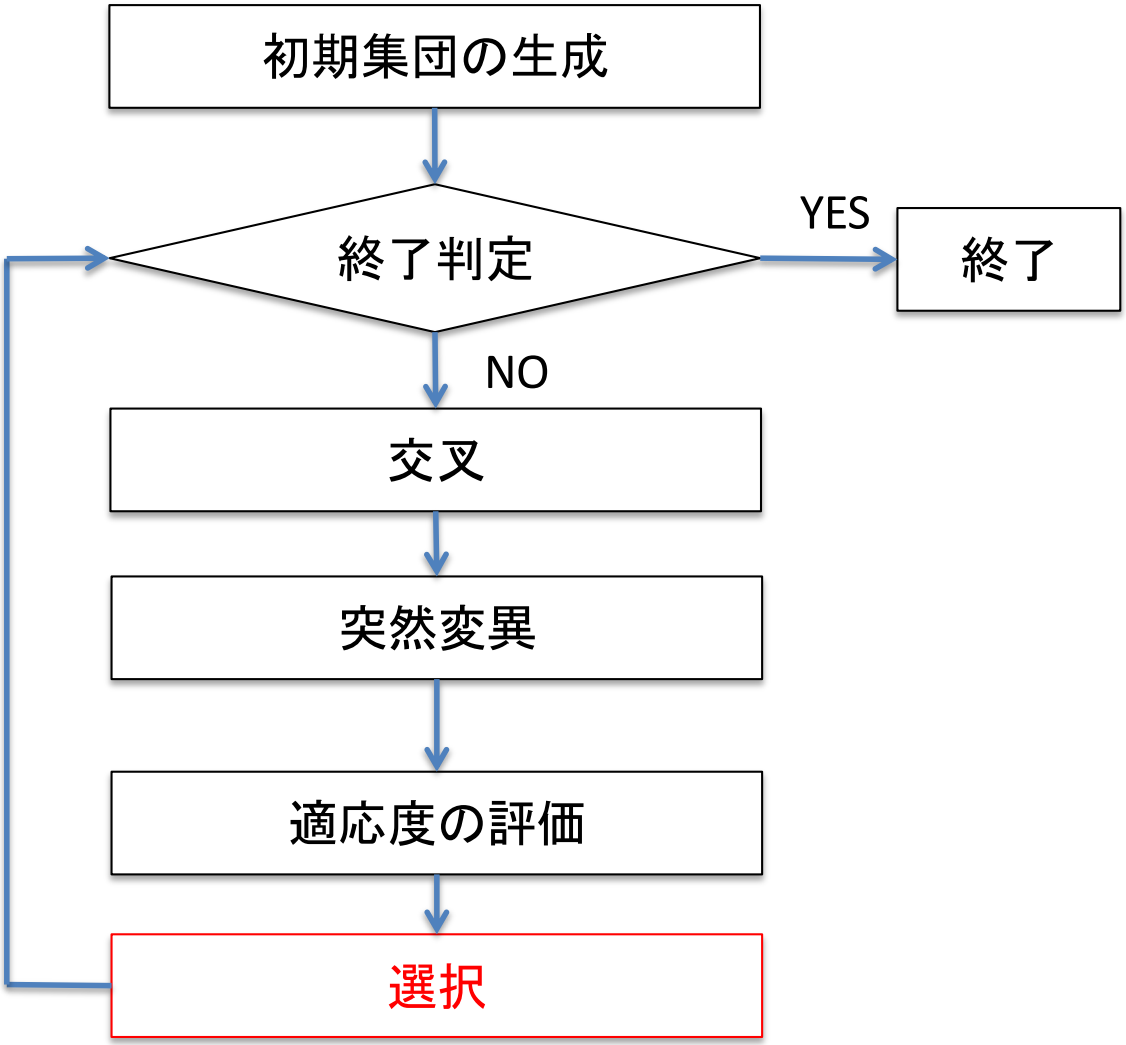
適応度 （fitness)

ある個体の子供の内の、繁殖年齢まで達したものの数

各個体の評価

親集団
1 1 1 0 0 1 … 0 1 0 1 1
0 1 0 0 1 1 … 1 0 0 1 0
0 0 1 1 1 0 … 1 1 0 0 0
+
子集団
0 0 1 0 1 0 … 1 1 0 1 0
0 1 0 1 1 1 … 1 0 0 0 0
0 1 1 1 1 0 … 1 1 0 0 0

ブランコの振れ幅に基づき各個体の適応度を計算


初期集団の生成
終了判定
YES
終了
NO
交叉
突然変異
適応度の評価
選択

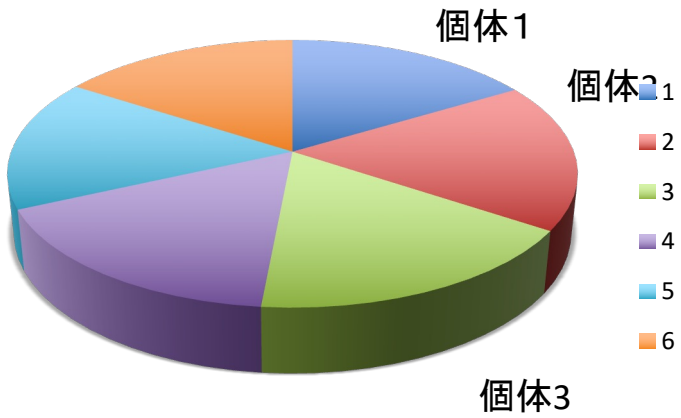
選択

親集団（サイズN)＋子集団(サイズM)から次世代の集団を構成するN個体を選ぶ

選択の方法は様々なものが考案されている
（１）ルーレット方式
（２）ランキング方式
（３）定常状態選択
（４）ボルツマン選択
（５）トーナメント方式
…

		適応度
個体1	1 1 1 0 0 1 … 0 1 0 1 1	**20**
個体2	0 1 0 0 1 1 … 1 0 0 1 0	**23**
個体3	0 0 1 1 1 0 … 1 1 0 0 0	**22**
個体a	0 0 1 0 1 0 … 1 1 0 1 0	**18**
個体b	0 1 0 1 1 1 … 1 0 0 0 0	**17**
個体c	0 1 1 1 1 0 … 1 1 0 0 0	**21**

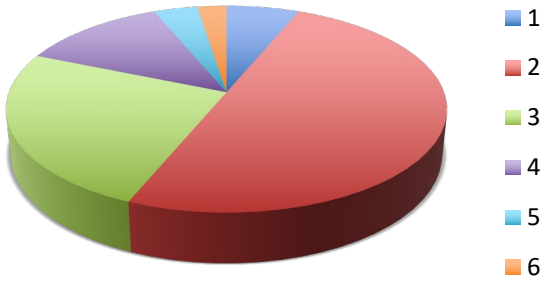
各個体をルーレット方式で選択する



ランキング方式

各個体を適応度によってランク付けして、「1位なら確率 p_1, 2位なら確率 p_2, 3位なら…」というふうにランクごとにあらかじめ確率を決めておく方式

		適応度	順位	確率
個体1	1 1 1 0 0 1 … 0 1 0 1 1	20	4	0.06
個体2	0 1 0 0 1 1 … 1 0 0 1 0	23	1	0.5
個体3	0 0 1 1 1 0 … 1 1 0 0 0	22	2	0.25
個体a	0 0 1 0 1 0 … 1 1 0 1 0	18	5	0.04
個体b	0 1 0 1 1 1 … 1 0 0 0 0	17	6	0.02
個体c	0 1 1 1 1 0 … 1 1 0 0 0	21	3	0.13



ルーレット方式の場合

（1）先の例の場合のように適応度にあまり差がつかない個体間に選択に関する差を導入

（2）最も適合度の高い個体がルーレットの90%を占める場合、他の個体は選ばれなくなる

エリート主義
(elitism)

最も高い適応度を示す個体あるいは複数の上位の適応度を示す個体を次世代に残し、残りをルーレット方式やランキング方式などで選択。

見つかった中で最も良いものを失わずにすむので、GAのパフォーマンスを向上させることができる

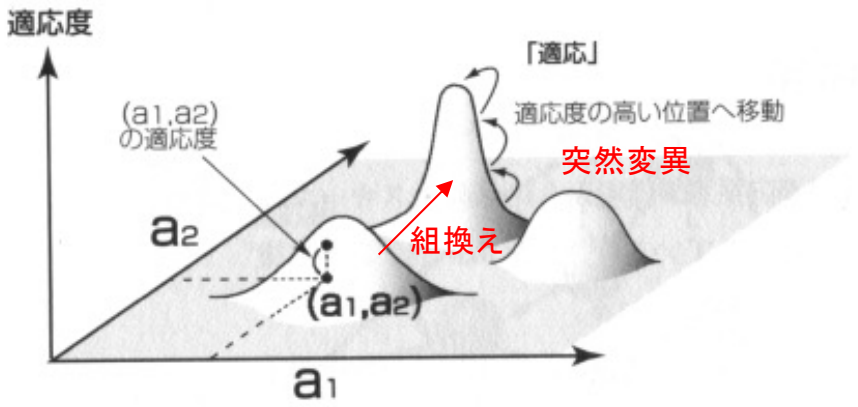
ここではランキング方式で、第二世代として以下が選択されたとする

- ０１００１１…１００１０　**23**　**1**　**0.5**
- ００１１１０…１１０００　**22**　**2**　**0.25**
- ０１００１１…１００１０　**23**　**1**　**0.5**

ここから終了判定移行し、終了と判定されなければ、第二世代について、
交叉、突然変異、選択を実施し、第三世代を作成する。

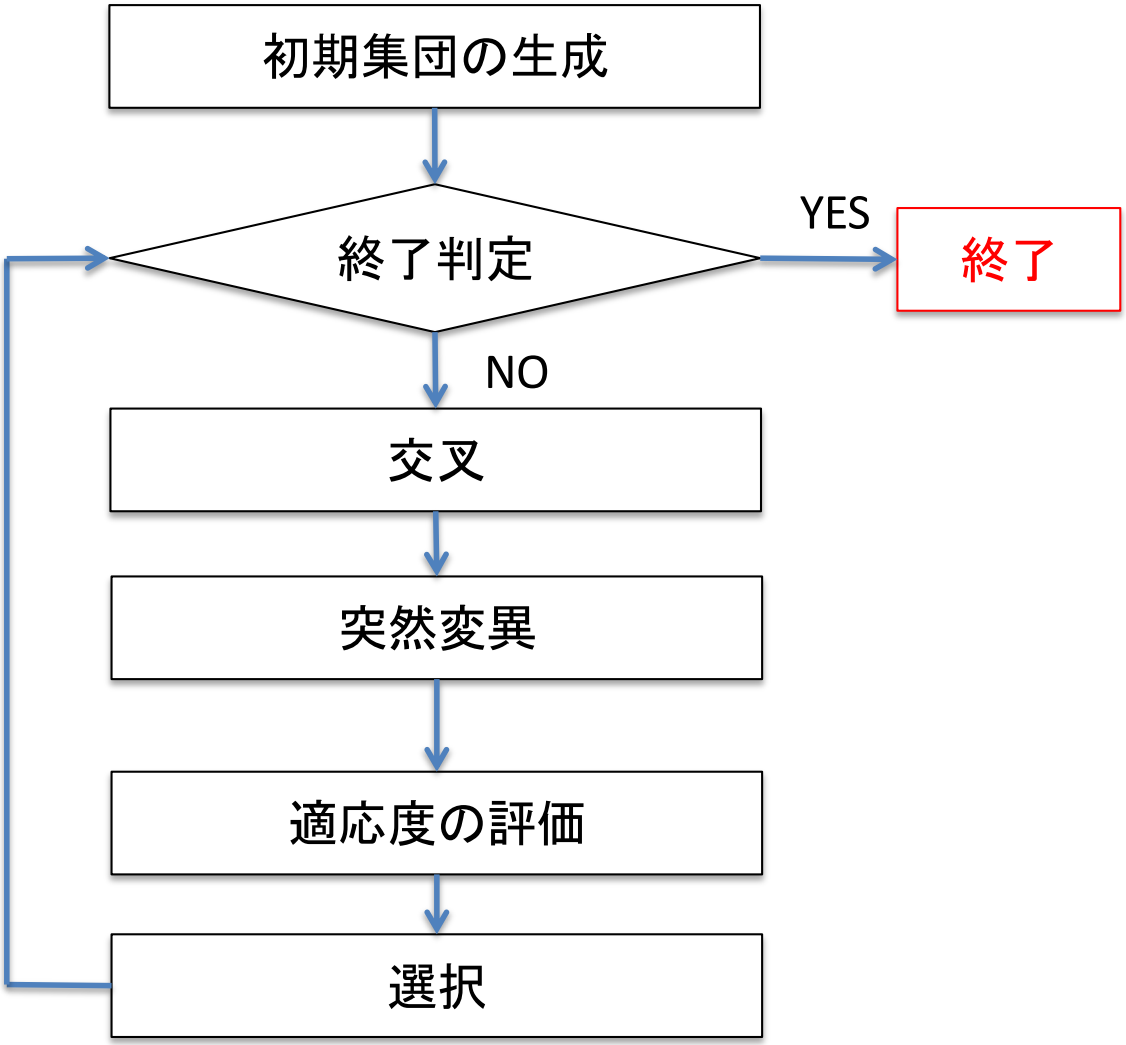
終了するまでこの操作を繰り返す。

適応度地形(フィットネス・ランドスケープ)で捉える「適応」(Adaptation)



「モデリングシミュレーション入門」
第13回 2005/01/14 井庭 崇 より

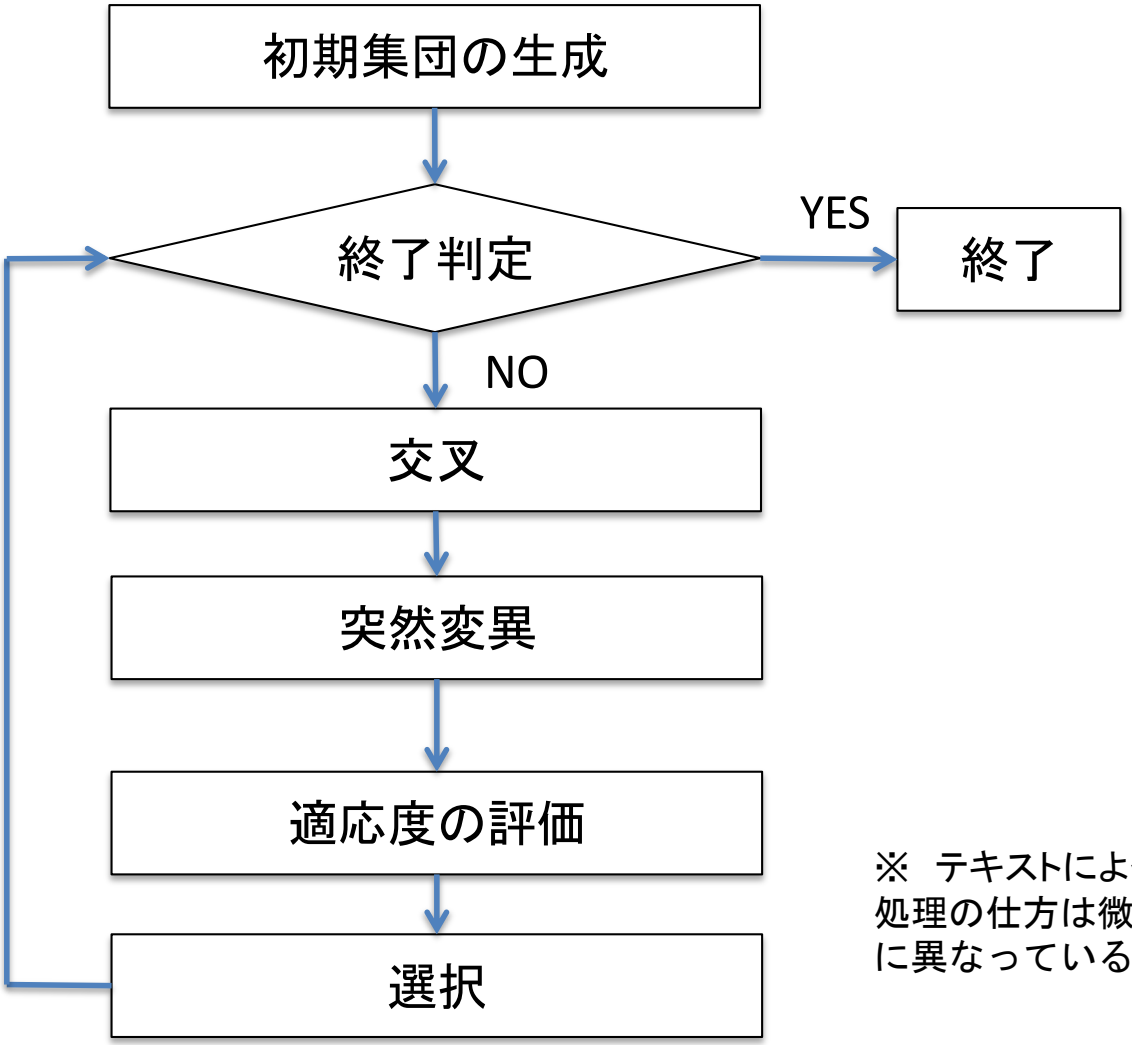
交叉によって、局所的な最適解に陥るのをふせぎ、大域的な最適解を探索できる。


初期集団の生成
終了判定
YES
終了
NO
交叉
突然変異
適応度の評価
選択

終了時処理

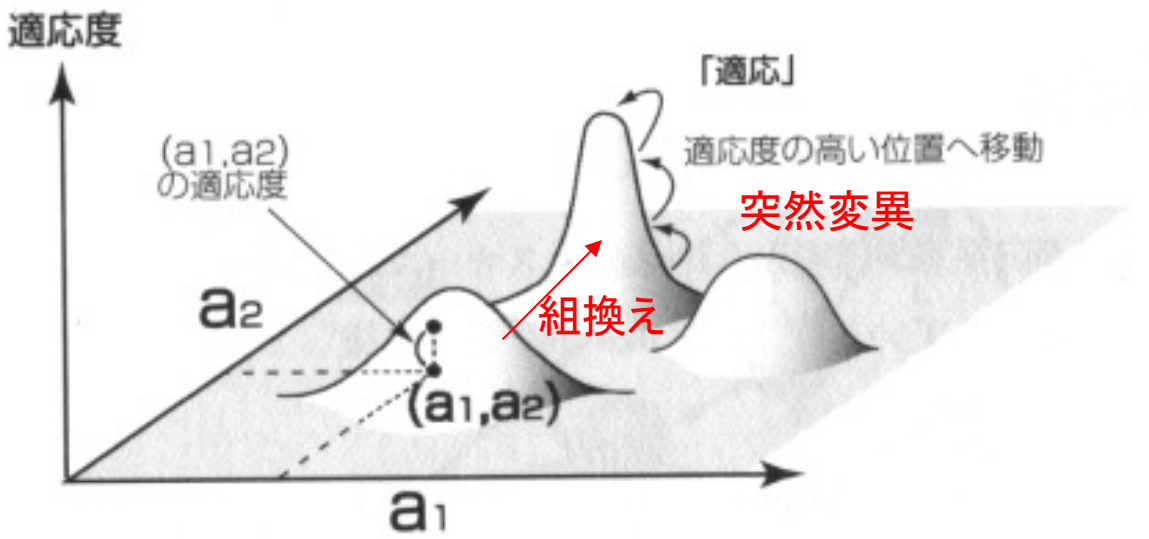
最終的に「現世代」の中で最も適応度の高い個体を「解」として出力する。

それまでの世代で生成された最も適応度の高い個体を「解」として出力する方法もある。



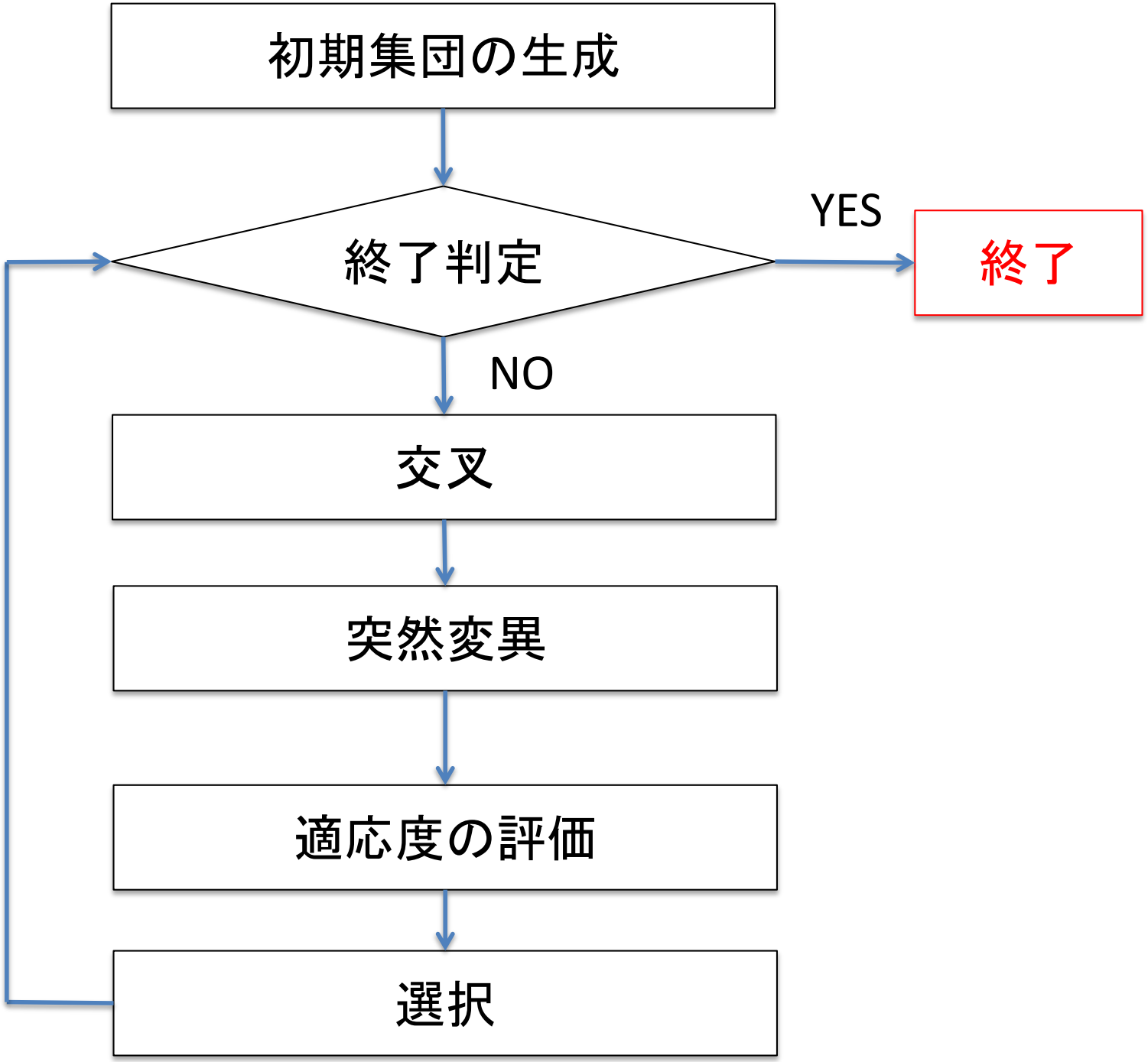
※　テキストによって
処理の仕方は微妙
に異なっている

適応度地形(フィットネス・ランドスケープ)で捉える「適応」(Adaptation)



「モデリングシミュレーション入門」
第13回 2005/01/14　井庭　崇　より

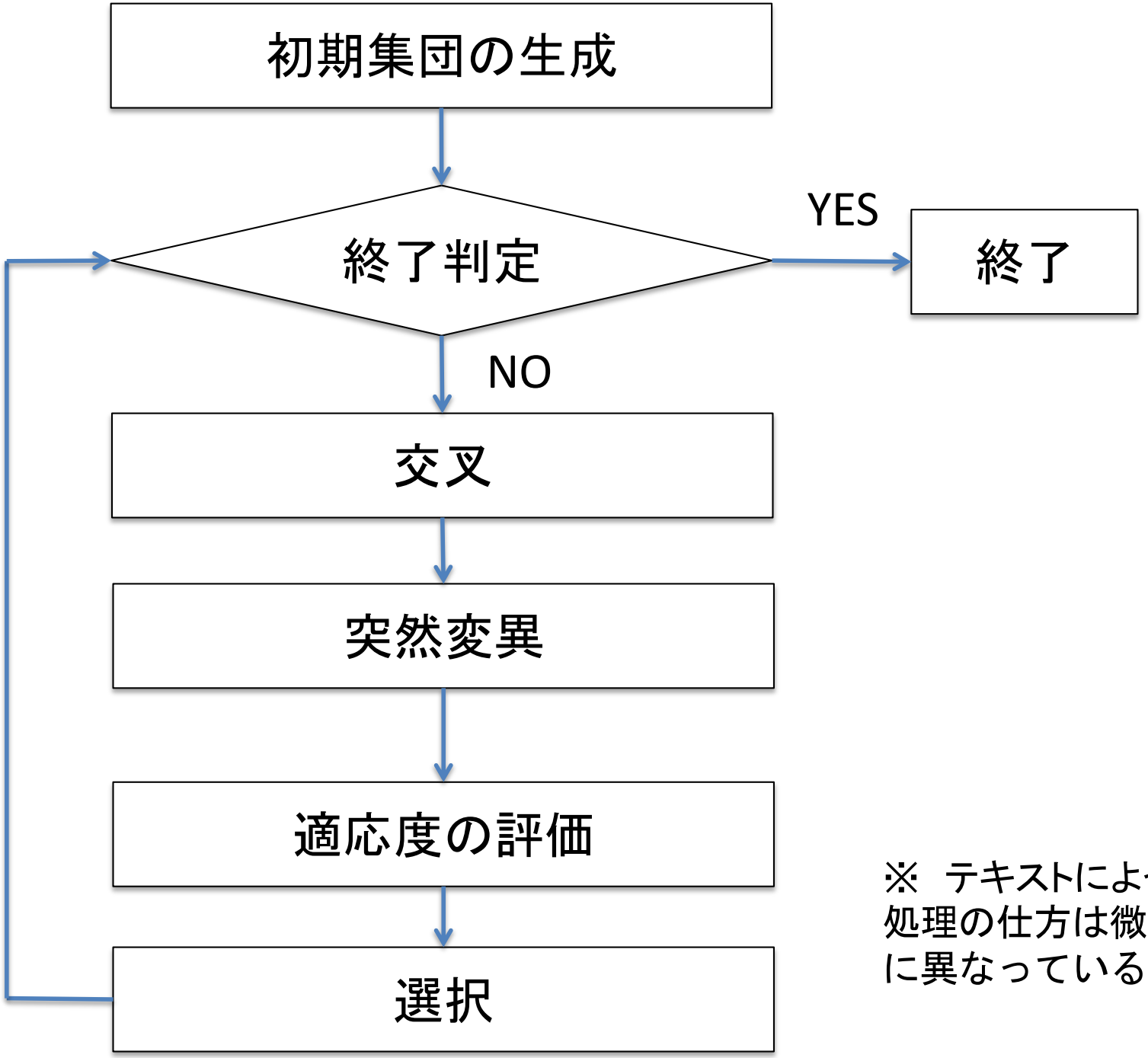
交叉によって、局所的な最適解に陥るのをふせぎ、大域的な最適解を探索できる。


初期集団の生成
終了判定
YES
終了
NO
交叉
突然変異
適応度の評価
選択

終了時処理

最終的に「現世代」の中で最も適応度の高い個体を「解」として出力する。

それまでの世代で生成された最も適応度の高い個体を「解」として出力する方法もある。



※　テキストによって
処理の仕方は微妙
に異なっている

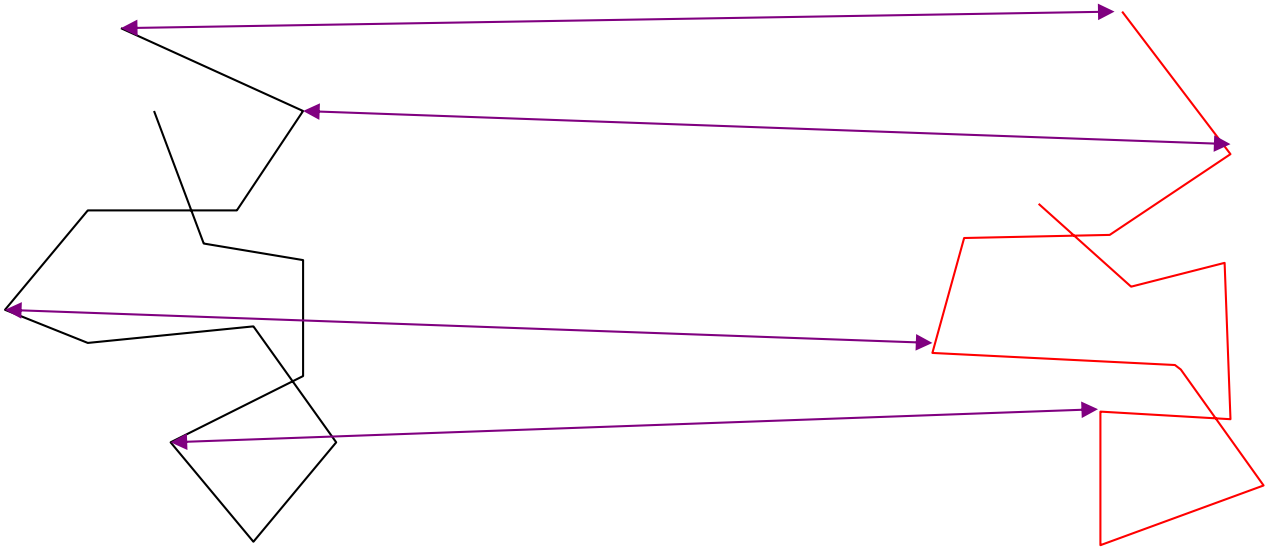
2. タンパク質の立体構造

2つの相同なタンパク質の立体構造の
重ね合わせを遺伝的アルゴリズムで行う

「相同」と「重ね合わせ」については後で説明

まず「タンパク質」について復習しよう

構造比較の原点　- 重ね合わせ (superposition) -

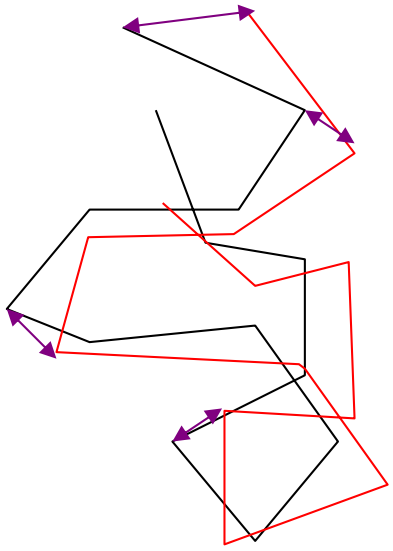


対応するCα原子間距離が最小になるように
二つの鎖を重ね合わせる (平行移動と回転)

McLachlan, A.D. (1972) *Nature New Biol*. 240, 83-85.

RMSD

rmsd (root mean square distance) = $\sqrt{\frac{1}{n}\sum(dist(A(i),B(i))^2}$



残基間対応が最初に
与えられていると
計算は容易

3.1.2 立体構造の計測法とそのデータ形式

生体高分子の立体構造データは，主に，X線結晶解析，核磁気共鳴法（NMR），電子顕微鏡の三つの方法で決められている（参考文献 1，2）．各手法の特徴を**表 3.1** にまとめた．生成された立体構造データの登録受付と配布は，米国の RCSB-PDB，欧州の PDBe，日本の PDBj（大阪大学の蛋白質研究所）の三つの拠点による共同組織 wwPDB（worldwide Protein Data Bank）が行っている．

表 3.1　生体高分子立体構造を決定する実験手法

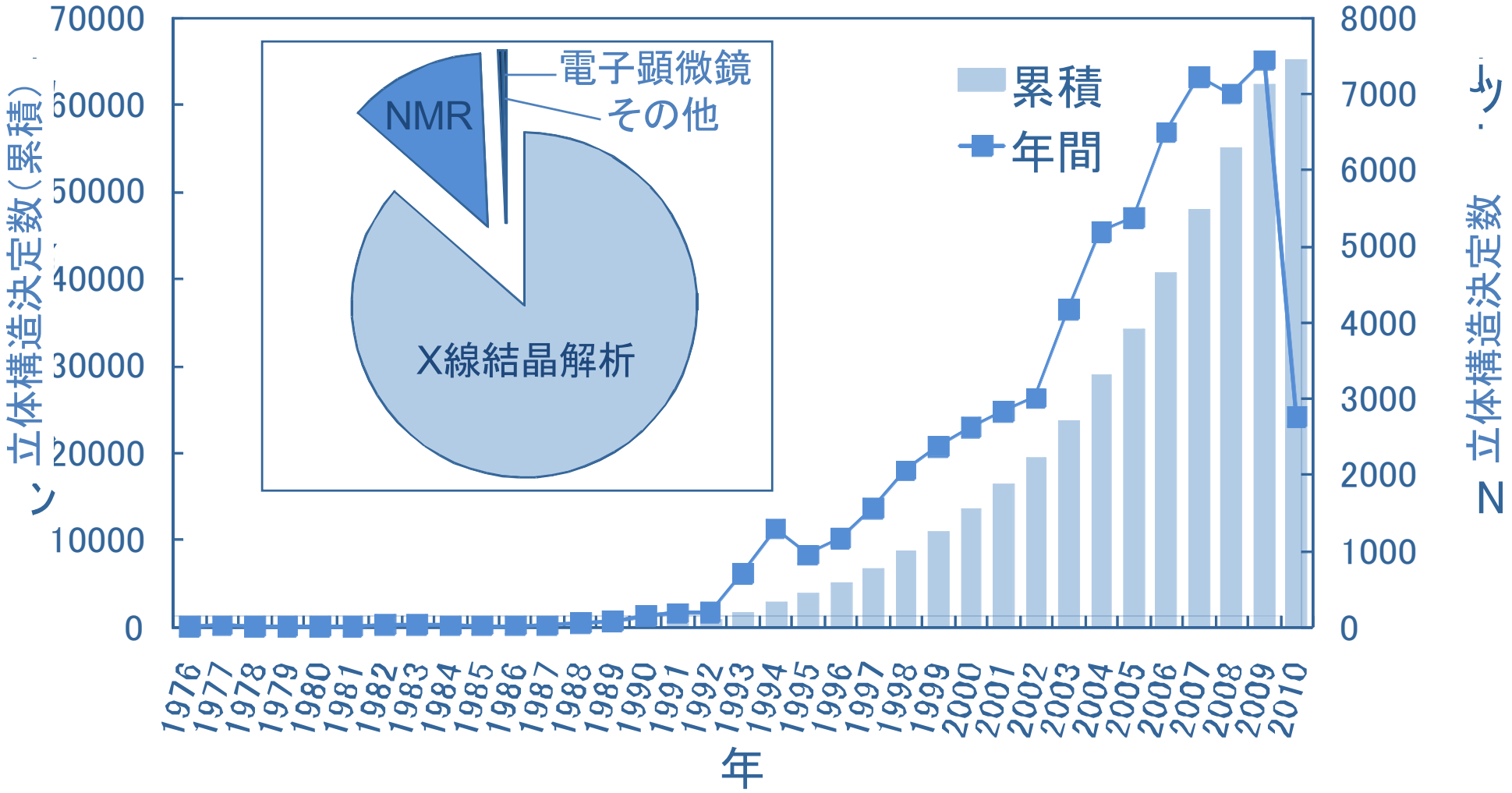
実験手法	X線結晶解析	核磁気共鳴（NMR）	電子顕微鏡
PDB データベースに占める割合※	89.5％	8.7％	1.6％
試料	結晶状態	同位体標識された高濃度の溶液状態	急速冷却によりグリッドに氷包埋された状態
データ測定	放射光などによるX線回折実験	NMR 装置によるシグナル測定	電子顕微鏡による 2D 画像群の撮影
計算過程	フーリエ逆変換・位相解決・原子モデルの構築	シグナル帰属，原子間距離の推定，原子モデルの構築	単粒子解析やトモグラフィーによる 3D マップ再構築，原子モデルの構築．
特徴	解像度は比較的高く，大きな分子量でも解析可能．解像度が特に高い場合を除き，水素原子は観測できない．	分子量の小さな分子の解析に向く．帰属に用いるため，水素原子の座標も記載されている．複数の候補モデルが書かれていることが多い．	比較的大きな分子の解析に向く．現状では中程度の解像度（3-5 Å）のデータが多い．

※ 2018 年 7 月 4 日の PDB の 141842 データの統計に基づく．

p.33-35

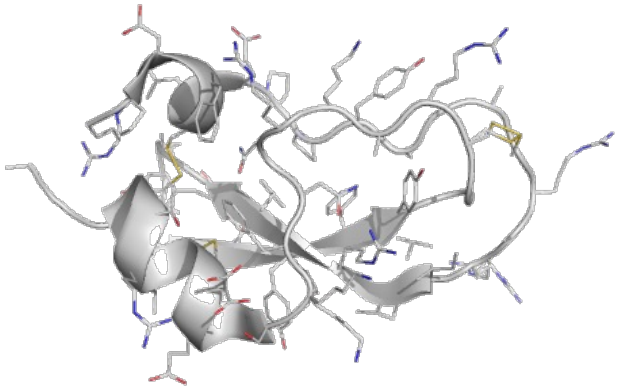
構造データの推移

- 立体構造情報の増加

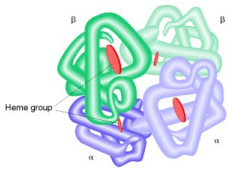


2010年5月4日付　(http://www.pdb.org/pdb/statistics/contentGrowthChart.do?content=total&seqid=100)

生体高分子の立体構造

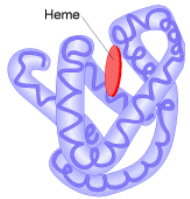
	遺伝子	タンパク質
配列	>1AUL gtctattagt actaatagac	>1QLQ:A RPDFCLEPPYAGACRARIIRYFYNA KAGLCQTFVYGGCRAKRNNFKSAED CLRTCGGA
	↓	折りたたみ （エネルギー的に安定） ↓
立体構造		

タンパク質立体構造の階層性



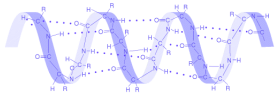
四次構造⟶

立体構造をとる二本以上のペプチド鎖から形成される構造
（具体例：ヘモグロビンは4つのペプチド鎖が集まり機能 (αサブユニット×２＋βサブユニット×２）

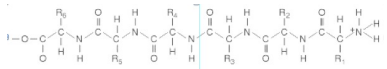


三次構造⟶

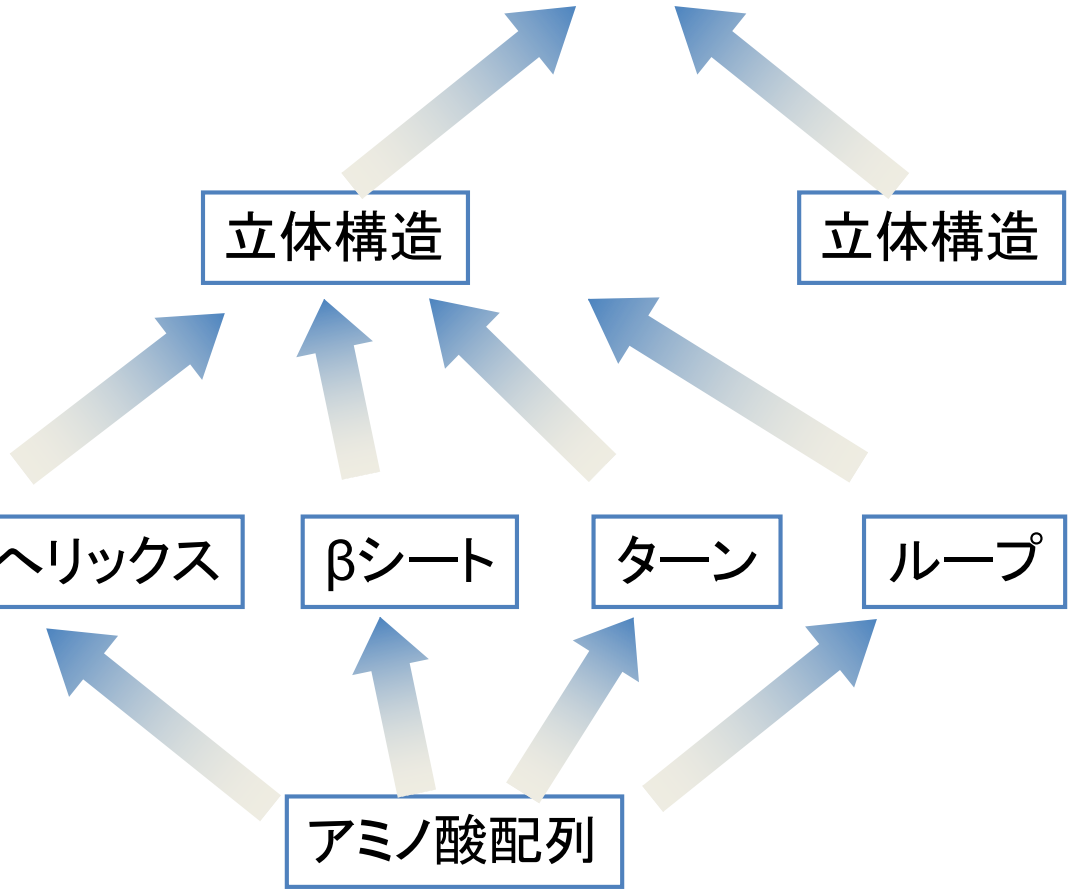
ドメイン構造 - -▶



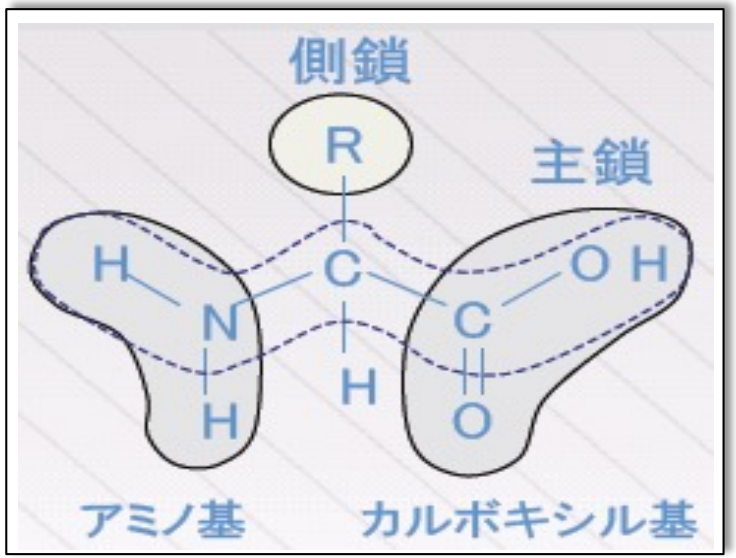
二次構造⟶

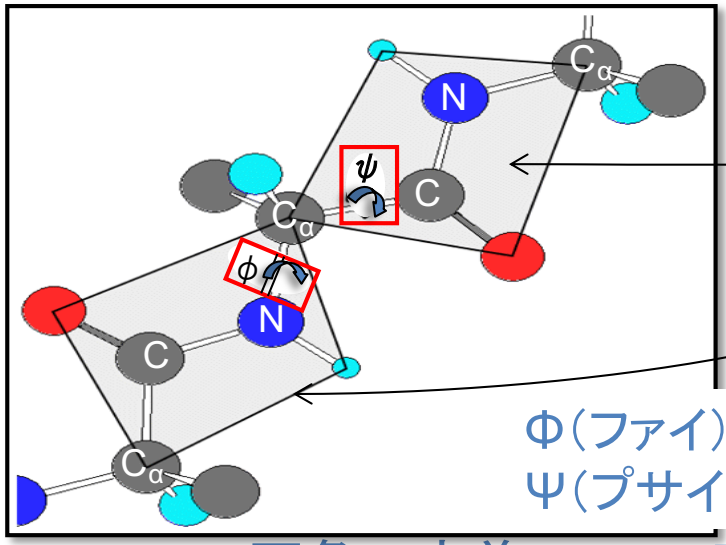


一次構造⟶



二面角（ねじれ角）





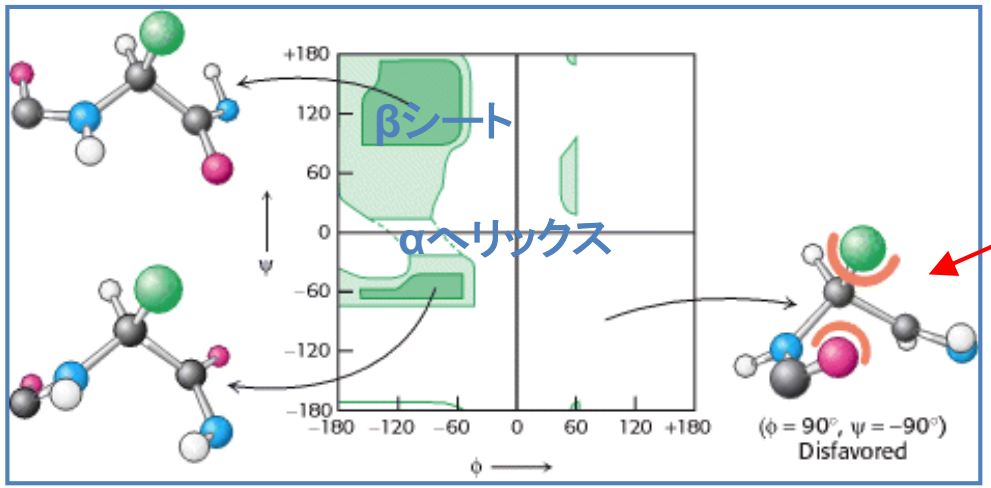
ペプチド結合は
平面構造をとる

Φ（ファイ）　:N-C_αの結合角
Ψ（プサイ）:C_α-Cの結合角

二面角の定義＝2つの面が形成する角度

http://www.ncbi.nlm.nih.gov/books/bv.fcgi?rid=bioinfo.figgrp.151

ラマチャンドランプロット



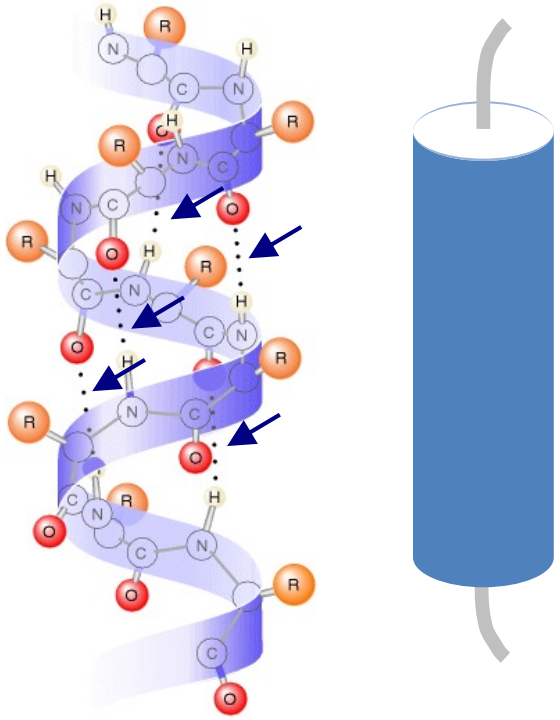
立体障害がおきる

二面角がとる値には
制限がある

http://www.ncbi.nlm.nih.gov/books/bv.fcgi?rid=stryer.figgrp.319

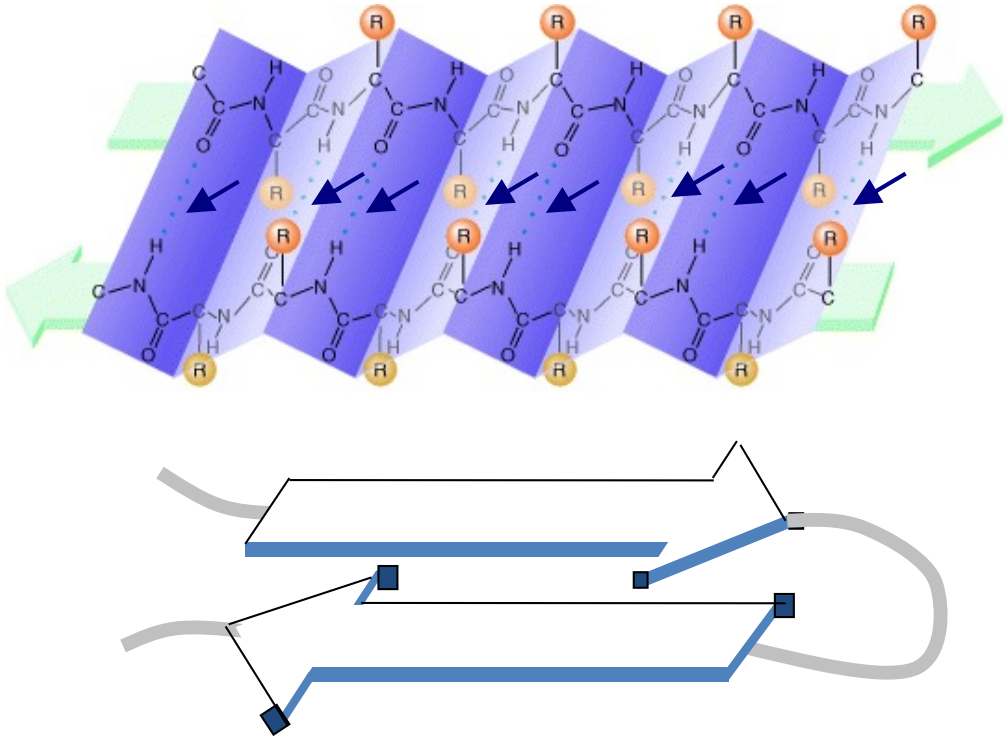
二次構造　規則的な水素結合のパターンを持つ

αヘリックス



4残基先のアミノ酸と水素結合を作る

βシート



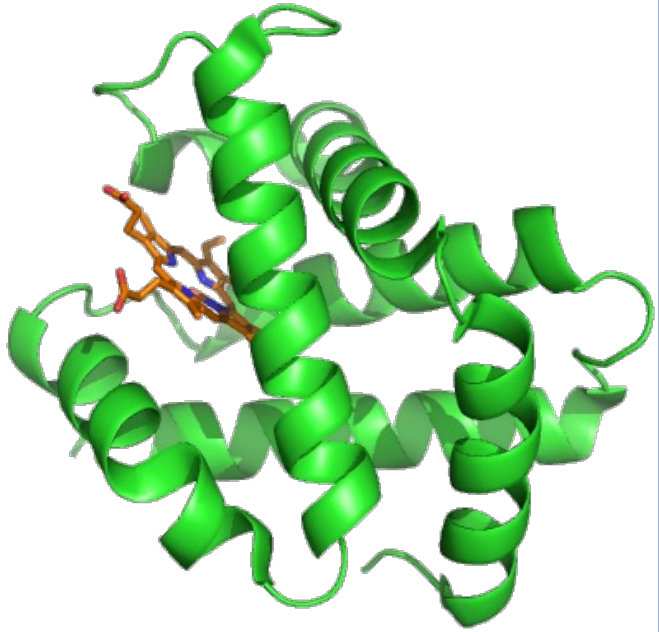
配列上離れた位置にアミノ酸と水素結合を作る

『MOLECULAR CELL BIOLOGY』より引用

三次構造

内側：
疎水性残基が集まる

外側：
親水性残基が多い



ヘモグロビンα鎖

独自の安定な構造をとる

ドメイン：

- 比較的安定な構造をとる単位
- 三次構造より（たいていの場合）小さな構造
- ドメインは、様々な三次構造中に見られる

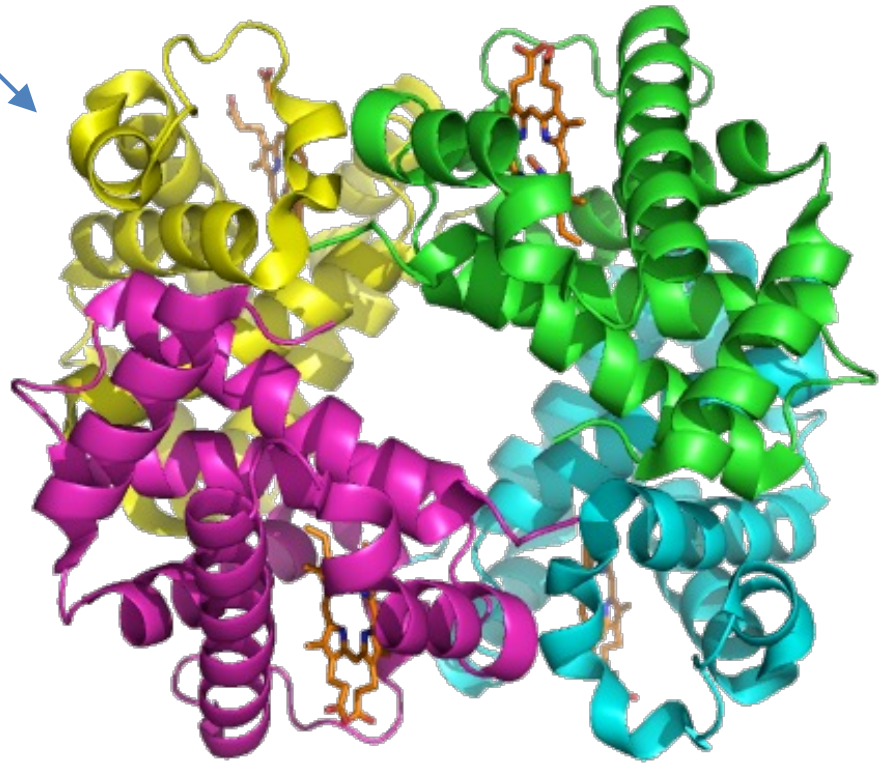
二次構造の組み合わせで、ある形に折りたたまれたもの。
一本のポリペプチドからなる。

立体構造保持に働く力

共有結合	ジスルフィド結合 （S-S結合）	アミノ酸配列中のシステイン残基が2個以上あり、それらが空間的に近い場合、イオウ原子同士が形成する結合。
非共有結合	イオン結合	反対の電荷を持つ残基間の電気的な結合。陽イオンと陰イオンがクーロン力で引き合う。
	水素結合	タンパク質分子内のアミノ基の水素が他のタンパク質分子内の酸素と電気的に強く引きあってできる結合。
	ファンデルワールス力	あらゆる分子同士で働く、互いに引き合う力。
	疎水結合	疎水性のアミノ酸残基の側鎖が、水を嫌って水と接触しないように集って生じる結合。

四次構造

サブユニット



ヘモグロビン（2W6V）

多量体の呼び方

1本：モノマー
2本：ダイマー
3本：トリマー
4本：テトラマー
5本：ペンタマー
6本：ヘキサマー

生体内で機能しているタンパク質の多くは、四次構造を形成している

複数のポリペプチドによって形成される立体構造

立体構造情報（座標データとして表現）

```
ATOM      3  C   MET A   1      26.913  26.639   3.531  1.00  9.62           C
ATOM      4  O   MET A   1      27.886  26.463   4.263  1.00  9.62           O
ATOM      5  CB  MET A   1      25.112  24.880   3.649  1.00 13.77           C
ATOM      6  CG  MET A   1      25.353  24.860   5.134  1.00 16.29           C
ATOM      7  SD  MET A   1      23.930  23.959   5.904  1.00 17.17           S
ATOM      8  CE  MET A   1      24.447  23.984   7.620  1.00 16.11           C
ATOM      9  N   GLN A   2      26.335  27.770   3.258  1.00  9.27           N
ATOM     10  CA  GLN A   2      26.850  29.021   3.898  1.00  9.07           C
ATOM     11  C   GLN A   2      26.100  29.253   5.202  1.00  8.72           C
ATOM     12  O   GLN A   2      24.865  29.024   5.330  1.00  8.22           O
ATOM     13  CB  GLN A   2      26.733  30.148   2.905  1.00 14.46           C
ATOM     14  CG  GLN A   2      26.882  31.546   3.409  1.00 17.01           C
ATOM     15  CD  GLN A   2      26.786  32.562   2.270  1.00 20.10           C
ATOM     16  OE1 GLN A   2      27.783  33.160   1.870  1.00 21.89           O
ATOM     17  NE2 GLN A   2      25.562  32.733   1.806  1.00 19.49           N
ATOM     18  N   ILE A   3      26.849  29.656   6.217  1.00  5.87           N
ATOM     19  CA  ILE A   3      26.235  30.058   7.497  1.00  5.07           C
ATOM     20  C   ILE A   3      26.882  31.428   7.862  1.00  4.01           C
ATOM     21  O   ILE A   3      27.906  31.711   7.264  1.00  4.61           O
ATOM     22  CB  ILE A   3      26.344  29.050   8.645  1.00  6.55           C
ATOM     23  CG1 ILE A   3      27.810  28.748   8.999  1.00  4.72           C
ATOM     24  CG2 ILE A   3      25.491  27.771   8.287  1.00  5.58           C
ATOM     25  CD1 ILE A   3      27.967  28.087  10.417  1.00 10.83           C
```

座標情報

文字列情報

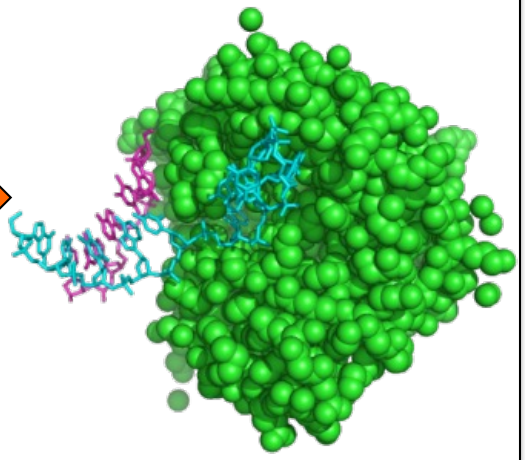
>2W36:F
cgatctgtagc （塩基配列）
>2W36:B
MDYRQLHRWDLPPEEAIKVQNELRKKIKLTPYEGEPEYVAGVALSFPGKEEGLAVIVVLEYPSFKILEVVSERGEITFPYIP
GLLAFREGPLFLKAWEKLRTKPDVVVFDGQGLAHPRKLGIASHMGLFIEIPTIGVAKSRLYGTFKMPEDKRCSWSYLYDGEE
IIGCVIRTKEGSAPIFVSPGHLMDVESSKRLIKAFTLPGRRIPEPTRLAHIYTQRLKKGLF （アミノ酸配列）

立体構造情報（座標データとして表現）

```
ATOM      3  C   MET A   1      26.913  26.639   3.531  1.00  9.62           C
ATOM      4  O   MET A   1      27.886  26.463   4.263  1.00  9.62           O
ATOM      5  CB  MET A   1      25.112  24.880   3.649  1.00 13.77           C
ATOM      6  CG  MET A   1      25.353  24.860   5.134  1.00 16.29           C
ATOM      7  SD  MET A   1      23.930  23.959   5.904  1.00 17.17           S
ATOM      8  CE  MET A   1      24.447  23.984   7.620  1.00 16.11           C
ATOM      9  N   GLN A   2      26.335  27.770   3.258  1.00  9.27           N
ATOM     10  CA  GLN A   2      26.850  29.021   3.898  1.00  9.07           C
ATOM     11  C   GLN A   2      26.100  29.253   5.202  1.00  8.72           C
ATOM     12  O   GLN A   2      24.865  29.024   5.330  1.00  8.22           O
ATOM     13  CB  GLN A   2      26.733  30.148   2.905  1.00 14.46           C
ATOM     14  CG  GLN A   2      26.882  31.546   3.409  1.00 17.01           C
ATOM     15  CD  GLN A   2      26.786  32.562   2.270  1.00 20.10           C
ATOM     16  OE1 GLN A   2      27.783  33.160   1.870  1.00 21.89           O
ATOM     17  NE2 GLN A   2      25.562  32.733   1.806  1.00 19.49           N
ATOM     18  N   ILE A   3      26.849  29.656   6.217  1.00  5.87           N
ATOM     19  CA  ILE A   3      26.235  30.058   7.497  1.00  5.07           C
ATOM     20  C   ILE A   3      26.882  31.428   7.862  1.00  4.01           C
ATOM     21  O   ILE A   3      27.906  31.711   7.264  1.00  4.61           O
ATOM     22  CB  ILE A   3      26.344  29.050   8.645  1.00  6.55           C
ATOM     23  CG1 ILE A   3      27.810  28.748   8.999  1.00  4.72           C
ATOM     24  CG2 ILE A   3      25.491  27.771   8.287  1.00  5.58           C
ATOM     25  CD1 ILE A   3      27.967  28.087  10.417  1.00 10.83           C
```

座標情報

可視化ソフト



立体構造

2w36(pdb)

文字列情報

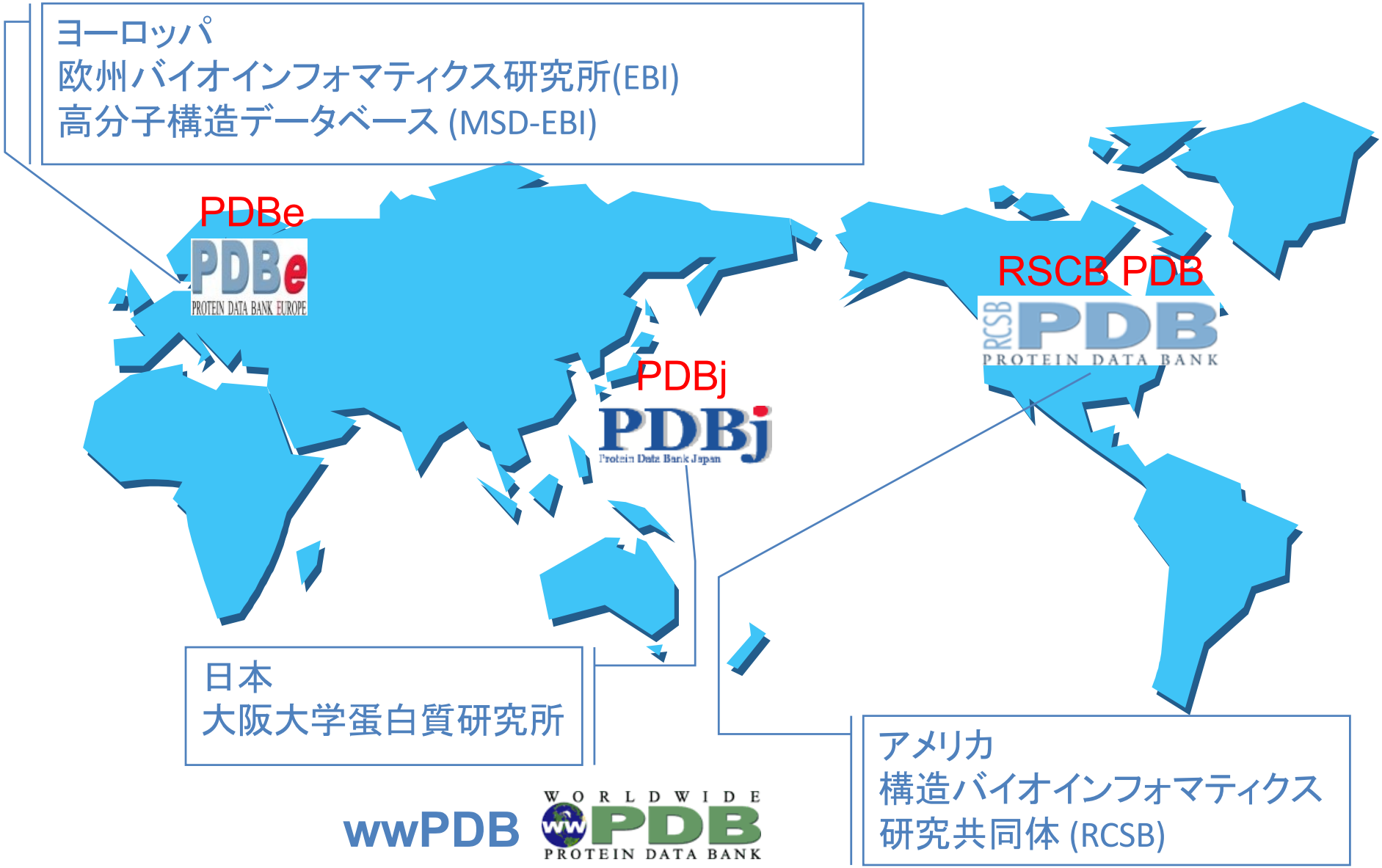
```
>2W36:F
cgatctgtagc
>2W36:B
MDYRQLHRWDLPPEEAIKVQNELRKKIKLTPYEGEPEYVAGVALSFPGKEEGLAVIVVLEYPSFKILEVVSERGEITFPYIP
GLLAFREGPLFLKAWEKLRTKPDVVVFDGQGLAHPRKLGIASHMGLFIEIPTIGVAKSRLYGTFKMPEDKRCSWSYLYDGEE
IIGCVIRTKEGSAPIFVSPGHLMDVESSKRLIKAFTLPGRRIPEPTRLAHIYTQRLKKGLF
```

(塩基配列)

(アミノ酸配列)

生体高分子の立体構造を扱ったデータベース

- 網羅的に立体構造情報を収集しているもの



生体高分子の立体構造を扱ったデータベース

- 網羅的に立体構造情報を収集しているもの

アメリカ
構造バイオインフォマティクス
研究共同体 (RCSB)

生体高分子の立体構造を扱ったデータベース

- PDBファイルの中身

```
HEADER    TRANSFERASE                             30-AUG-04   1XBB
TITLE     CRYSTAL STRUCTURE OF THE SYK TYROSINE KINASE DOMAIN WITH
TITLE    2 GLEEVEC
COMPND    MOL_ID: 1;
COMPND   2 MOLECULE: TYROSINE-PROTEIN KINASE SYK;
COMPND   3 CHAIN: A;
COMPND   4 SYNONYM: SPLEEN TYROSINE KINASE;
COMPND   5 EC: 2.7.1.112;
COMPND   6 ENGINEERED: YES
SOURCE    MOL_ID: 1;
SOURCE   2 ORGANISM_SCIENTIFIC: HOMO SAPIENS;
SOURCE   3 ORGANISM_COMMON: HUMAN;
SOURCE   4 ORGANISM_TAXID: 9606;
SOURCE   5 GENE: SYK;
SOURCE   6 EXPRESSION_SYSTEM: SPODOPTERA FRUGIPERDA;
SOURCE   7 EXPRESSION_SYSTEM_COMMON: FALL ARMYWORM;
SOURCE   8 EXPRESSION_SYSTEM_TAXID: 7108;
SOURCE   9 EXPRESSION_SYSTEM_CELL_LINE: SF9;
SOURCE  10 EXPRESSION_SYSTEM_VECTOR_TYPE: BACULOVIRUS
KEYWDS    GLEEVEC, STI-571, IMATINIB, SYK, SPLEEN TYPROSINE KINASE,
KEYWDS   2 ACTIVE CONFORMATION, STRUCTURAL GENOMICS, STRUCTURAL
KEYWDS   3 GENOMIX, TRANSFERASE
EXPDTA    X-RAY DIFFRACTION
AUTHOR    V.L.NIENABER,S.ATWELL,J.M.ADAMS,J.BADGER,M.D.BUCHANAN,
AUTHOR   2 I.K.FEIL,K.J.FRONING,X.GAO,J.HENDLE,K.KEEGAN,B.C.LEON,
AUTHOR   3 H.J.MULLER-DEICKMANN,B.W.NOLAND,K.POST,K.R.RAJASHANKAR,
AUTHOR   4 A.RAMOS,M.RUSSELL,S.K.BURLEY,S.G.BUCHANAN
 （省略）
```

生体高分子の立体構造を扱ったデータベース

- PDBファイルの中身

```
HEADER     TRANSFERASE
TITLE      CRYSTAL STRUCT
TITLE    2 GLEEVEC
COMPND     MOL_ID: 1;
COMPND    2 MOLECULE: TYR
COMPND    3 CHAIN: A;
COMPND    4 SYNONYM: SPLE
COMPND    5 EC: 2.7.1.112
COMPND    6 ENGINEERED: Y
SOURCE     MOL_ID: 1;
SOURCE    2 ORGANISM_SCIE
SOURCE    3 ORGANISM_COMM
SOURCE    4 ORGANISM_TAXI
SOURCE    5 GENE: SYK;
SOURCE    6 EXPRESSION_SY
SOURCE    7 EXPRESSION_SY
SOURCE    8 EXPRESSION_SY
SOURCE    9 EXPRESSION_SY
SOURCE   10 EXPRESSION_SY
KEYWDS     GLEEVEC, STI-5
KEYWDS    2 ACTIVE CONFOR
KEYWDS    3 GENOMIX, TRAN
EXPDTA     X-RAY DIFFRACT
AUTHOR     V.L.NIENABER,S
AUTHOR    2 I.K.FEIL,K.J.
AUTHOR    3 H.J.MULLER-DE
AUTHOR    4 A.RAMOS,M.RUS
 （省略）
```

ヘッダー	ボディ
HEADER	物質の分類名、日付、ファイル名
COMPND	タンパク質名
SOURCE	物質が由来する生物名
AUTHOR	座標を作成した著者名
REVDAT	データが登録された日
JRNL	文献情報
REMARK	実験条件などに関する情報
SEQRES	アミノ酸配列
HET	アミノ酸以外の原子
HELIX	α―ヘリックスに関する情報
SHEET	β―シートに関する情報
TURN	β―ターンに関する情報
SSBOND	ジスルフィド結合に関する情報
ATOM	個々のアミノ酸の原子の座標
HETATM	アミノ酸以外の原子の座標
CONTACT	原子の結合に関する情報
END	ファイルの終了

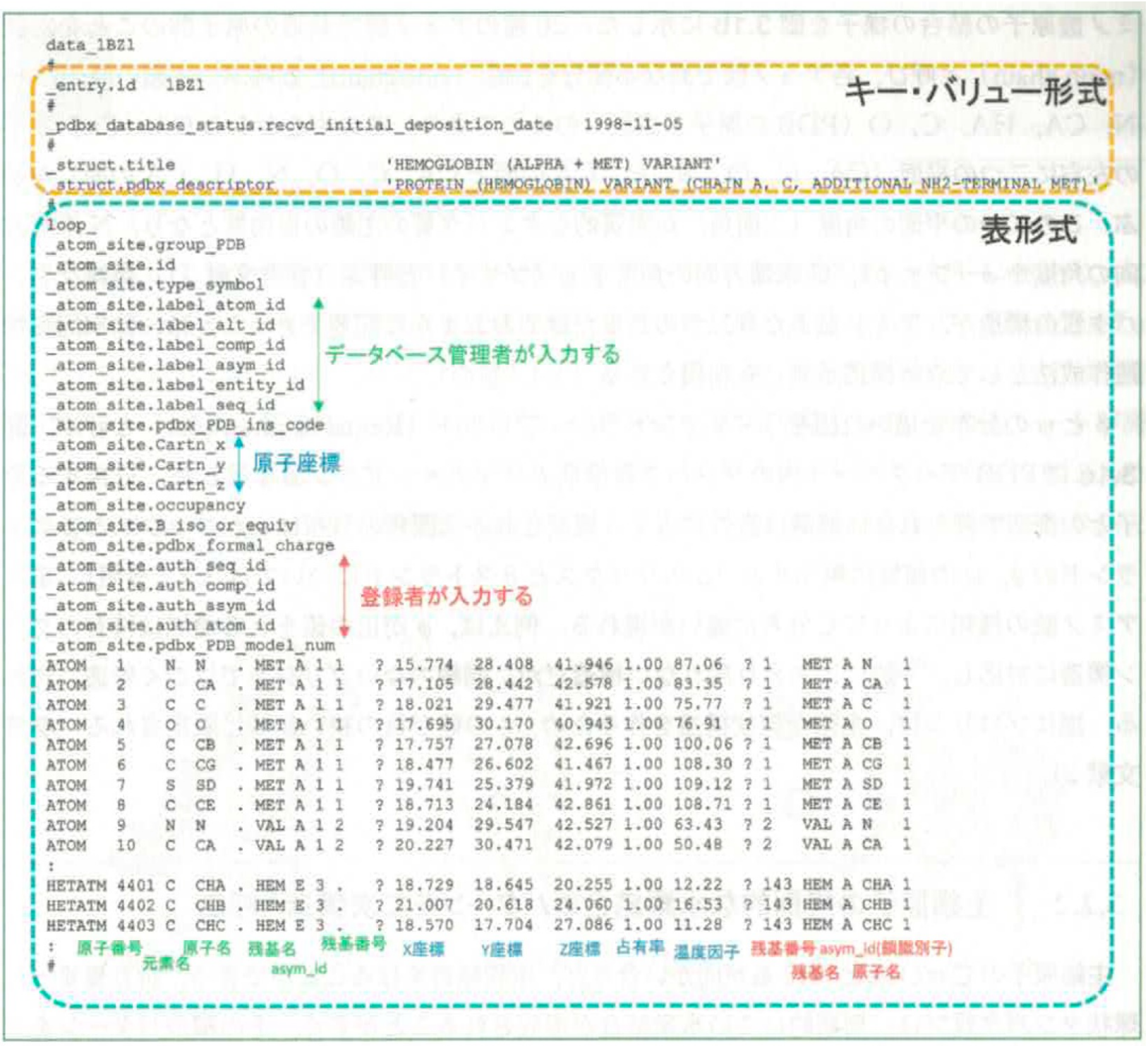
生体高分子の立体構造を扱ったデータベース

- **座標データ**

次に，立体構造データがどういう形式でデータベースに格納されるのかを説明する．まず，各立体構造データには，“1mbd”“4hhb”といった「1文字の数字＋3文字の英数字」のPDB IDコードがついている．各データは，立体構造が計測された一つのまとまりが単位であり，一つのタンパク質鎖が必ずしも一つのPDB IDに対応するわけではない．タンパク質の一部だけが入っている場合，複数のタンパク質が含まれている場合もある．DNAやRNAなどの高分子や，結合している低分子化合物の立体構造も含まれている．**図3.1a**にヒト・ヘモグロビンの立体構造（PDB ID：1bz1）をリボンモデルで示した．このデータには4つのタンパク質鎖と4つのヘム分子が入っている．**図3.1b**には最初の三つのアミノ酸の原子構造を示した．この立体構造に対応するデータを，旧PDBフォーマット（**図3.2**）とmmCIFフォーマット（**図3.3**）の二つの形式で示した．どちらも，立体構造は，1原子が1行に記載され，原子の中心点のXYZ座標の値が書かれている．座標値の単位はÅ（オングストローム）である（1 Å＝0.1 nm＝10^{-10} m）．また，各原子には原子番号，残基番号，3文字以下の残基名（例えば，MET，ALA）と4文字以下の原子名（CA，N，C，O，CBなど）が割り当てられている．3文字の残基名は，アミノ酸以外の分子にも割り当てられている．例えば，ヘム分子は“HEM”，ATP分子は“ATP”，カルシウムイオンは“CA”，イレッサは“IRE”などと決められている．一つのデータに複数の分子が入っている場合には，それぞれ，A，B，Cといった鎖識別子（chain identifier）がついている．

p.33-34

旧PDBフォーマットと2014年から標準フォーマットとなったmmCIFフォーマットの違いについて簡単に説明しよう．1970年代から使われていた旧PDBフォーマットは，わかりやすい反面，1行80文字の固定幅であるため，原子番号，残基番号，鎖識別子の文字数に制限が設定され，記載できる原子数，鎖数に明らかな上限がある．例えば，鎖識別子は1文字であるため，小文字や数字を使っても，タンパク質鎖数が100を超える巨大分子は記載できないことになる．そこで導入されたのが，mmCIFフォーマットである．このフォーマットは，スペースで区切られた形式を基本としているため，原子番号や鎖識別子の文字数に制限はない．また，登録者が入力する残基番号などの情報とデータベース管理者が入力する情報の両方を記載することで利便性と統一性を実現している．現在PDBでは，大部分のデータでは旧PDBフォーマットでもダウンロードできるが，一部の巨大分子の立体構造（例えば，リボソーム：4v42，光化学系Ⅱ：4v62など）は旧PDBフォーマットでは記載できないため，mmCIFフォーマットだけで公開されている．




```
data_1BZ1
#
_entry.id   1BZ1
#
_pdbx_database_status.recvd_initial_deposition_date   1998-11-05
#
_struct.title                     'HEMOGLOBIN (ALPHA + MET) VARIANT'
_struct.pdbx_descriptor           'PROTEIN (HEMOGLOBIN) VARIANT (CHAIN A, C, ADDITIONAL NH2-TERMINAL MET)'
#
loop_
_atom_site.group_PDB
_atom_site.id
_atom_site.type_symbol
_atom_site.label_atom_id
_atom_site.label_alt_id
_atom_site.label_comp_id
_atom_site.label_asym_id
_atom_site.label_entity_id
_atom_site.label_seq_id
_atom_site.pdbx_PDB_ins_code
_atom_site.Cartn_x
_atom_site.Cartn_y
_atom_site.Cartn_z
_atom_site.occupancy
_atom_site.B_iso_or_equiv
_atom_site.pdbx_formal_charge
_atom_site.auth_seq_id
_atom_site.auth_comp_id
_atom_site.auth_asym_id
_atom_site.auth_atom_id
_atom_site.pdbx_PDB_model_num
ATOM   1    N  N   . MET A 1 1   ? 15.774  28.408  41.946 1.00 87.06  ? 1   MET A N   1
ATOM   2    C  CA  . MET A 1 1   ? 17.105  28.442  42.578 1.00 83.35  ? 1   MET A CA  1
ATOM   3    C  C   . MET A 1 1   ? 18.021  29.477  41.921 1.00 75.77  ? 1   MET A C   1
ATOM   4    O  O   . MET A 1 1   ? 17.695  30.170  40.943 1.00 76.91  ? 1   MET A O   1
ATOM   5    C  CB  . MET A 1 1   ? 17.757  27.078  42.696 1.00 100.06 ? 1   MET A CB  1
ATOM   6    C  CG  . MET A 1 1   ? 18.477  26.602  41.467 1.00 108.30 ? 1   MET A CG  1
ATOM   7    S  SD  . MET A 1 1   ? 19.741  25.379  41.972 1.00 109.12 ? 1   MET A SD  1
ATOM   8    C  CE  . MET A 1 1   ? 18.713  24.184  42.861 1.00 108.71 ? 1   MET A CE  1
ATOM   9    N  N   . VAL A 1 2   ? 19.204  29.547  42.527 1.00 63.43  ? 2   VAL A N   1
ATOM   10   C  CA  . VAL A 1 2   ? 20.227  30.471  42.079 1.00 50.48  ? 2   VAL A CA  1
:
HETATM 4401 C  CHA . HEM E 3 .   ? 18.729  18.645  20.255 1.00 12.22  ? 143 HEM A CHA 1
HETATM 4402 C  CHB . HEM E 3 .   ? 21.007  20.618  24.060 1.00 16.53  ? 143 HEM A CHB 1
HETATM 4403 C  CHC . HEM E 3 .   ? 18.570  17.704  27.086 1.00 11.28  ? 143 HEM A CHC 1
:
#
```

図 3.3　mmCIF フォーマットによる立体構造データの例

正式には PDBx/mmCIF フォーマットという．PDB ID：1bz1 の，いくつかの項目と，原子座標の一部を示す．このフォーマットは，項目名と値のペアを記述するシンプルな「キー・バリュー形式」（オレンジの点線枠内）と，項目名を最初に示しデータが行として多数並ぶ「表形式」（青色の点線枠内）のどちらかで記述されている．値の区切り文字はスペースで，値の長さは可変である．項目名はカテゴリ名とアイテム名がピリオド（.）で組み合わされている．例えば，_atom_site.Cartn_x では，atom_site がカテゴリ名，Cartn_x がアイテム名である．表形式の場合，カテゴリ名は表の名前，アイテム名は列の名前に相当する．atom_site の表が，旧 PDB フォーマットの ATOM 行と HETATM 行に相当する．残基番号（seq_id），残基名（comp_id），鎖識別子（asym_id），原子名（atom_id）は，データベース管理者が入力する label_ で始まる項目（緑色）と登録者が入力する auth_ で始まる項目（赤色）の両方が入っている．これらはまったく同一である場合も多いが，異なることも少なくない．この例では，HETATM 行の HEM の asym_id は，データベース入力の label_asym_id は E だが，登録者入力の auth_asym_id は A となっている．データベース管理者は，このヘム分子は 5 つ目の分子であるから E としたが，登録者はタンパク質の A 鎖に結合している分子だから A としたのだと考えられる．

演習

- ヒトの“チロシンキナーゼ（SYK：spleen tyrosine kinase）”の座標データを入手する

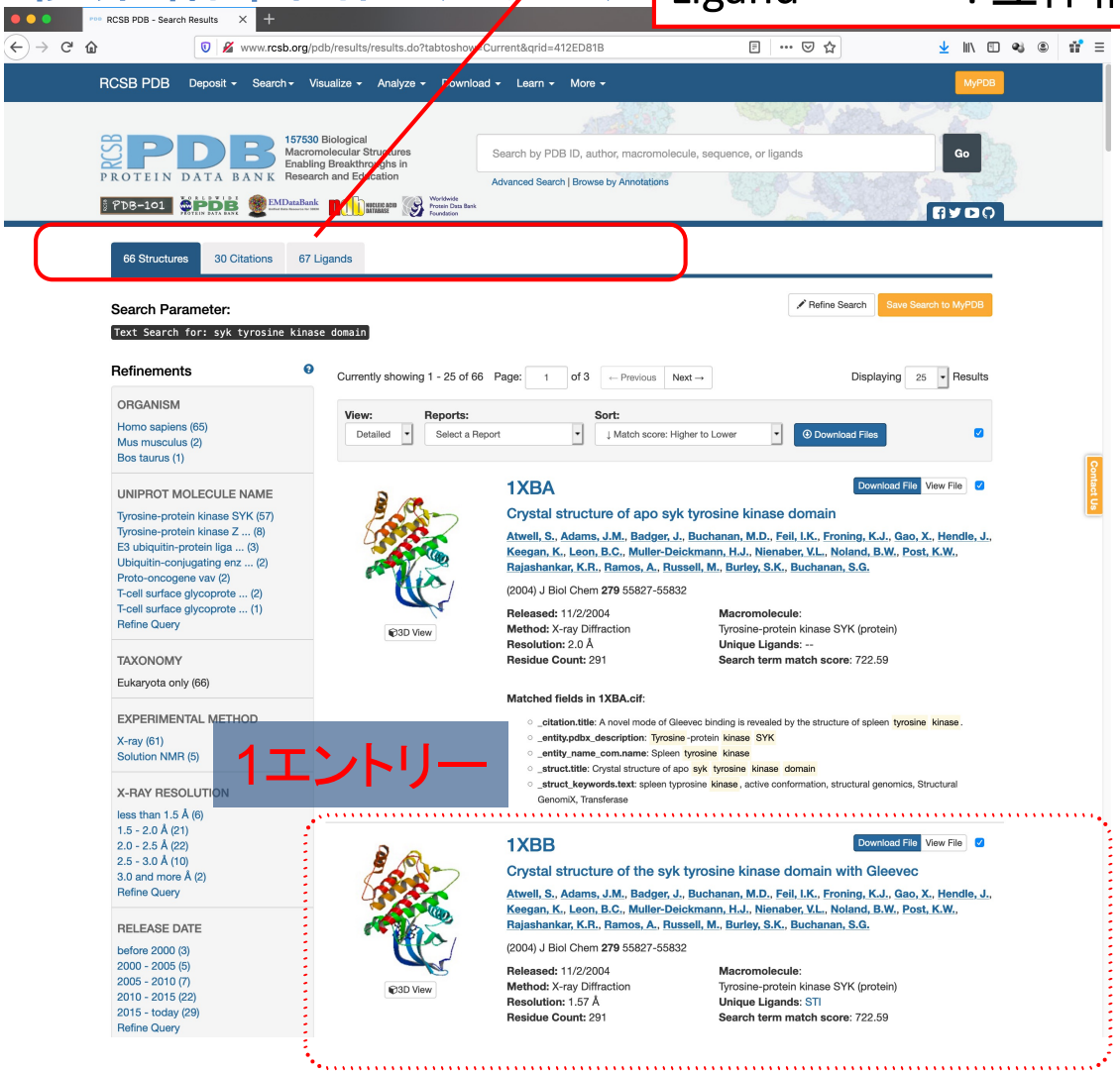
立体構造情報を手に入れる

RCSB PDBのトップページ

立体構造情報を手に入れる

Structures　：立体構造をリスト
Citations　：立体構造に関する文献をリスト
Ligand　：立体構造中に含まれるリガンドをリスト

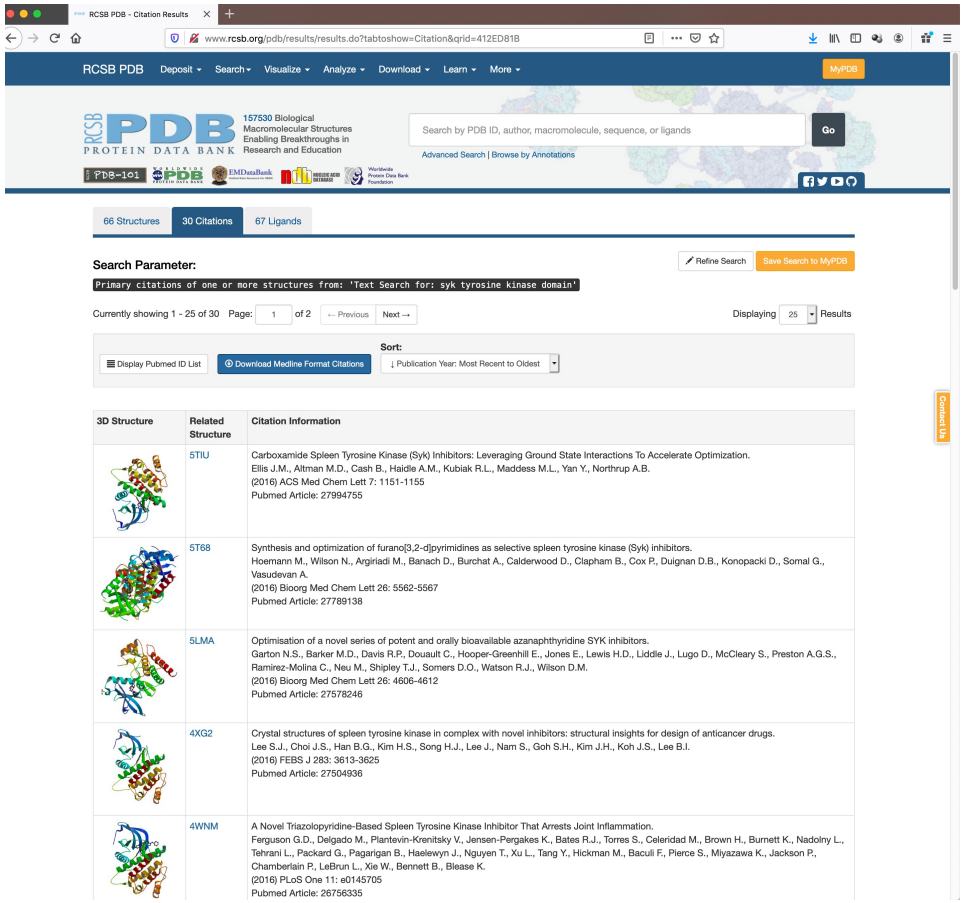
検索結果画面（トップ）

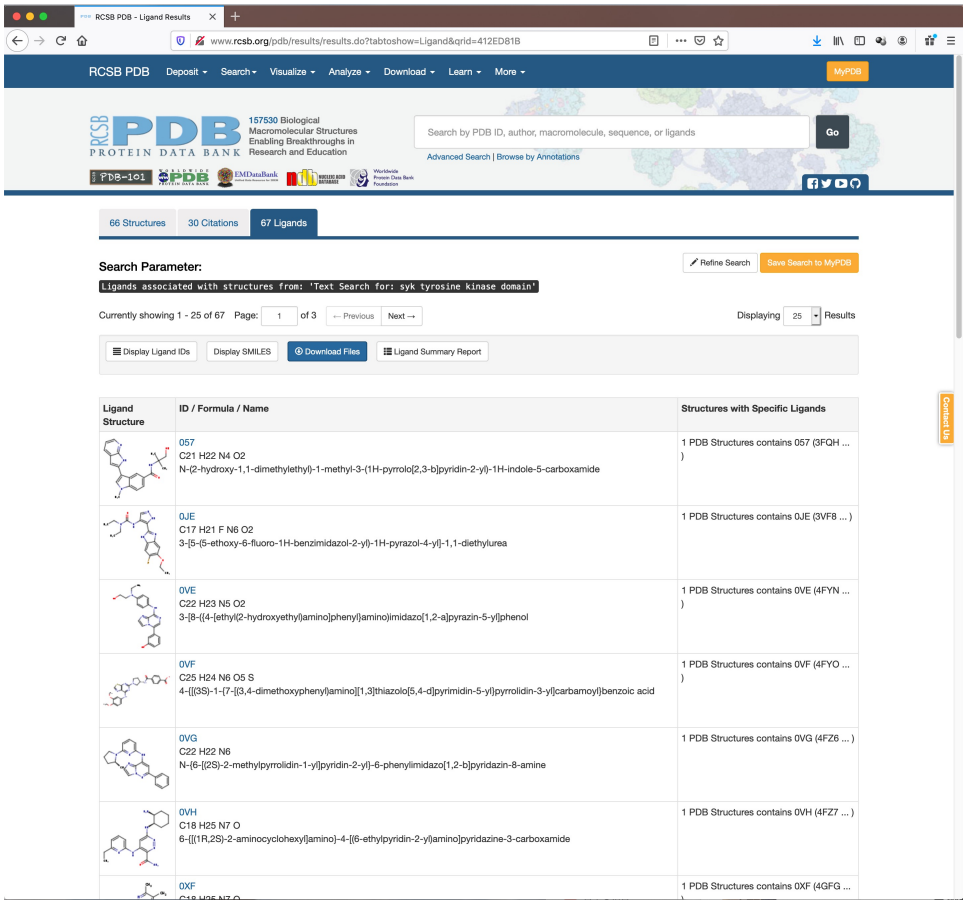


66構造ヒット

Structure タブがデフォルトで示される

立体構造情報を手に入れる



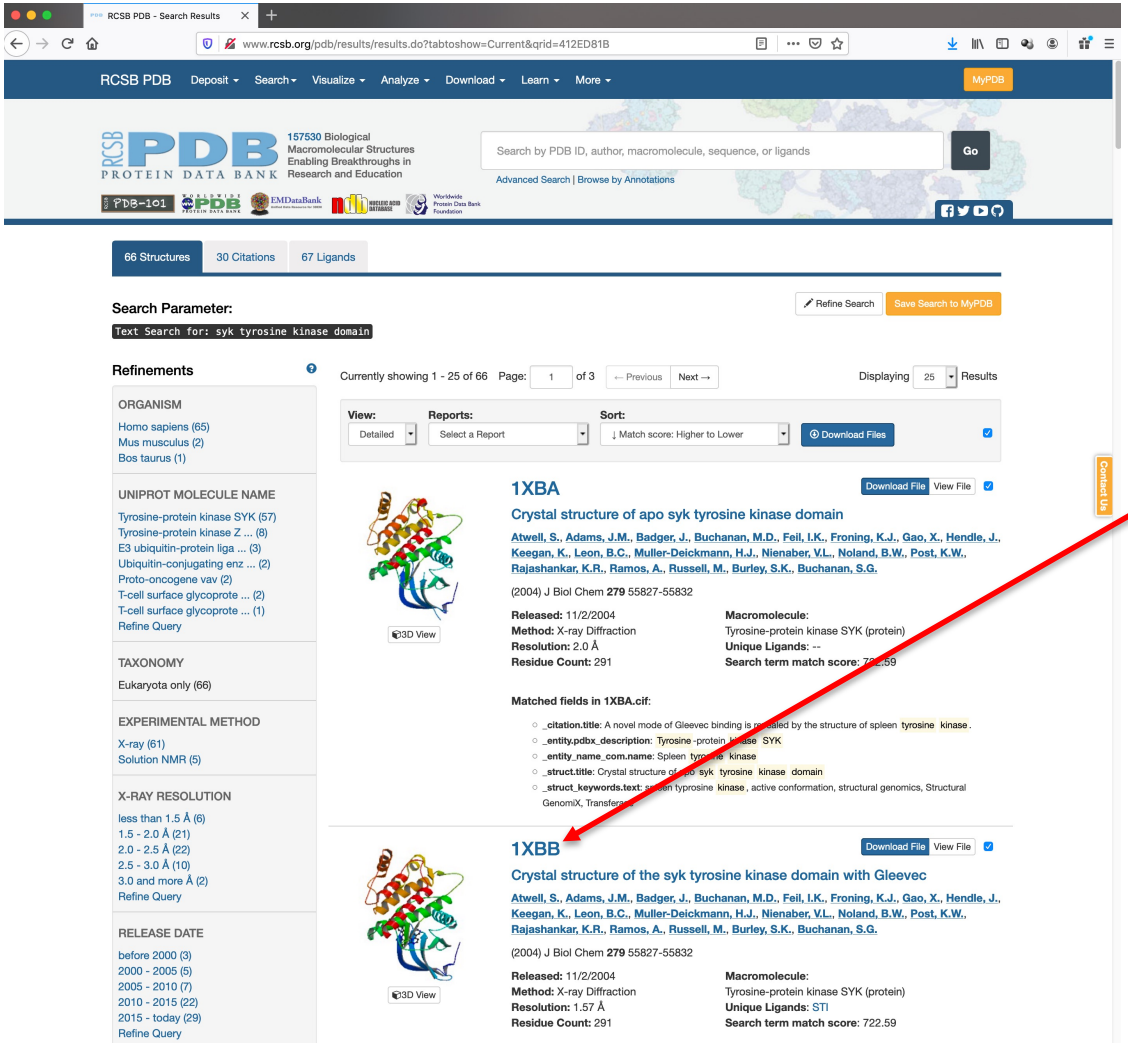


Citation タブ

Ligandタブ

Structure タブに戻す

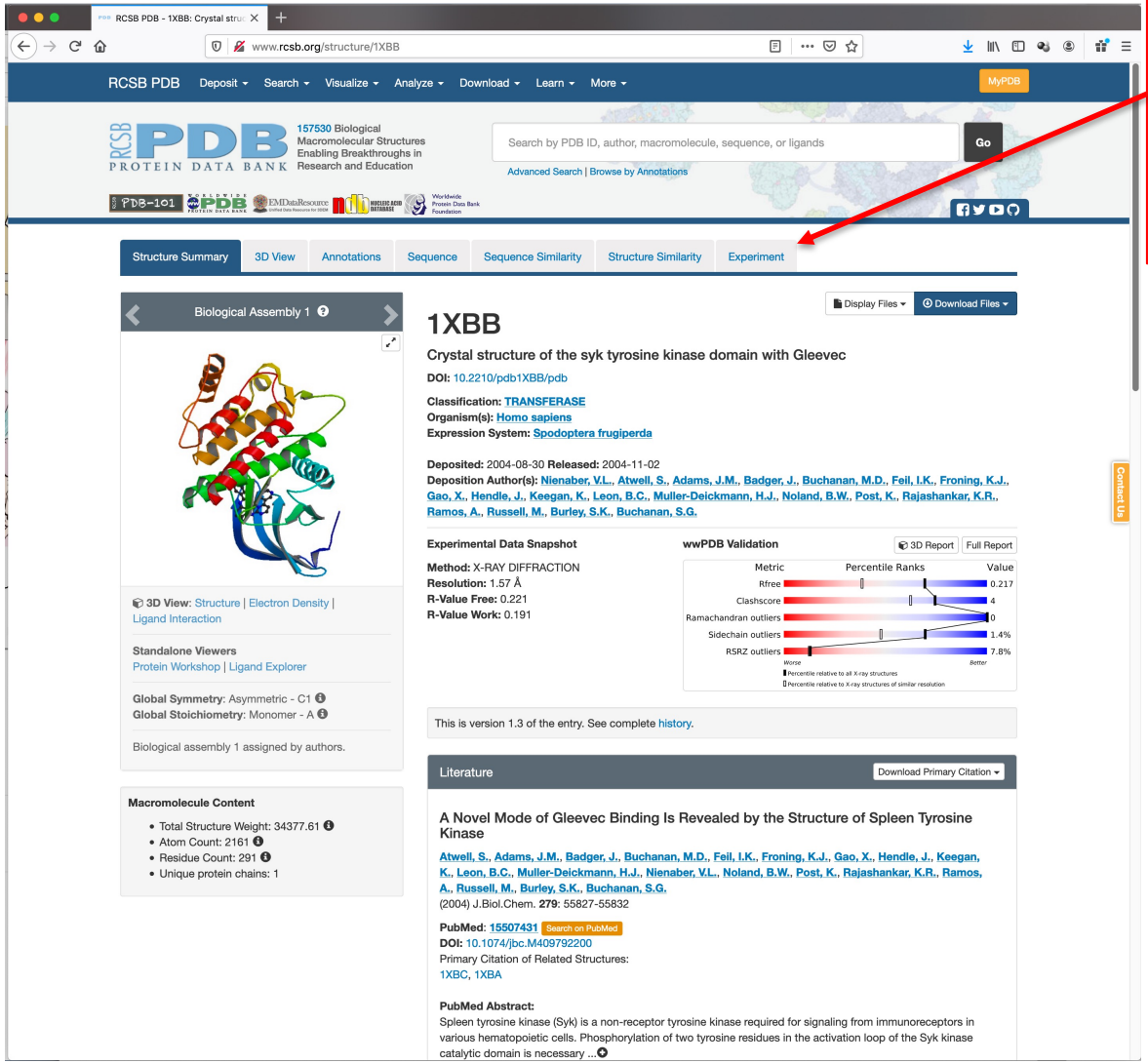
立体構造情報を手に入れる



1XBBをクリック

立体構造情報を手に入れる

エントリ：1XBBのデータ画面（トップ）



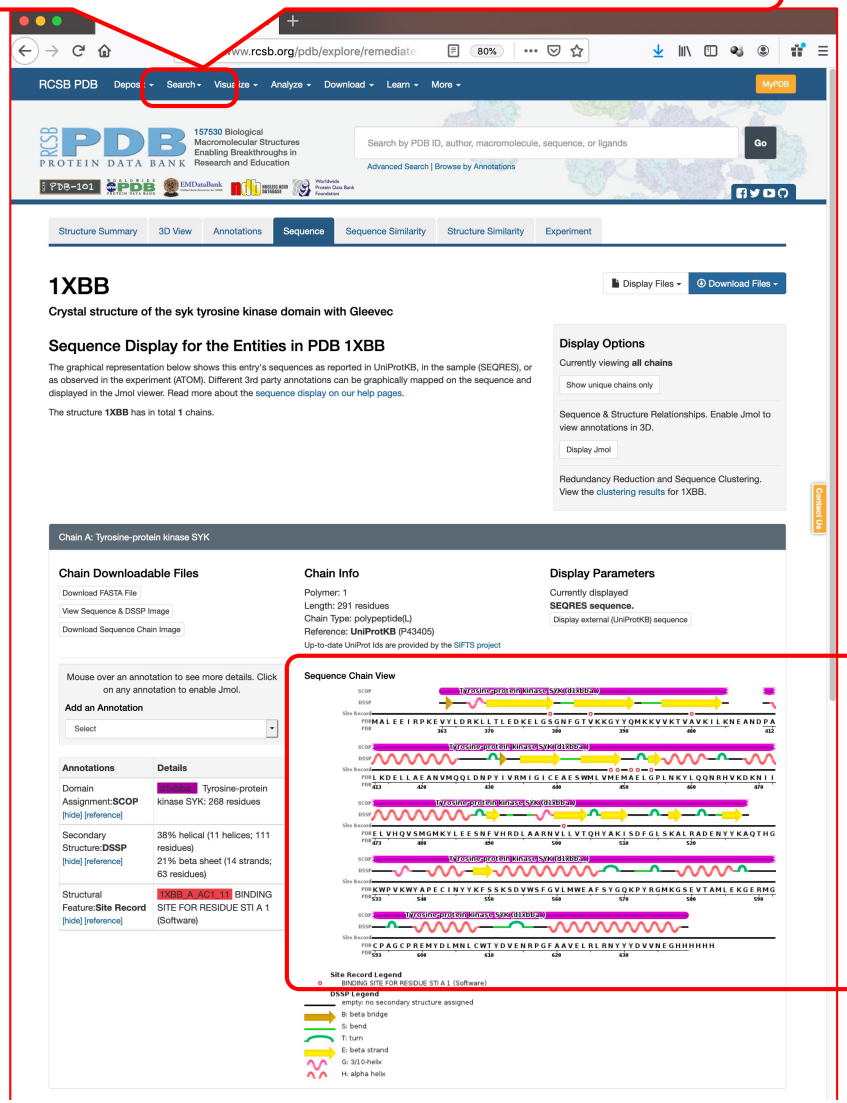
Structure Summary　：エントリーの概要
3D View　：構造グラフィクス
Annotation　:ドメイン、ファミリーの注釈付
Sequence　：配列の特徴
Seq. Similarity　：類似した配列の情報
Str. Similarity　：類似した立体構造の情報
Experiment　：構造決定の実験条件

デフォルトはStructure Summary

立体構造情報を手に入れる

エントリ：1XBBのデータ画面（ページ上部のタブをクリックすると。。。）

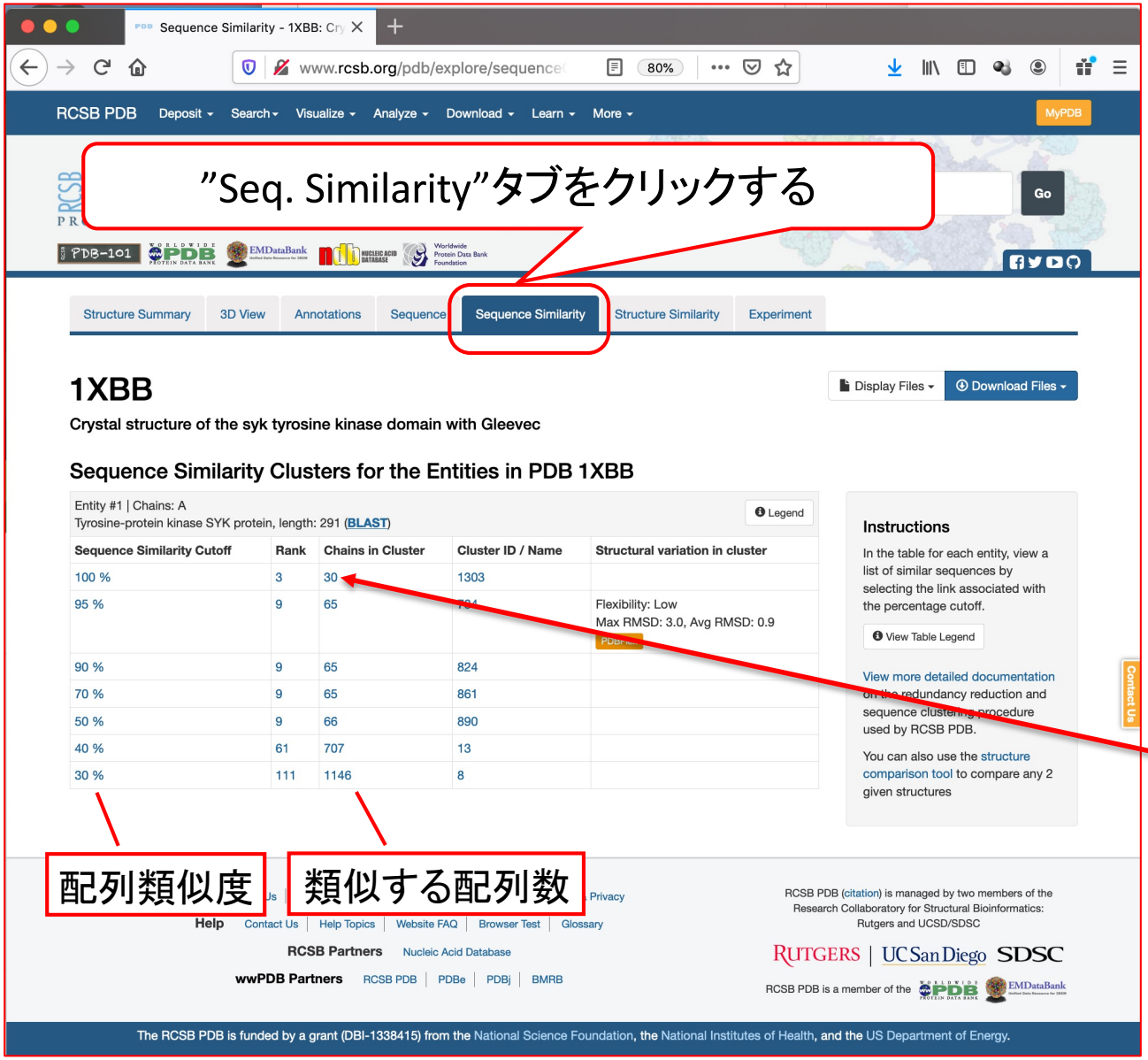
”Sequence”タブをクリックする



二次構造情報

立体構造情報を手に入れる

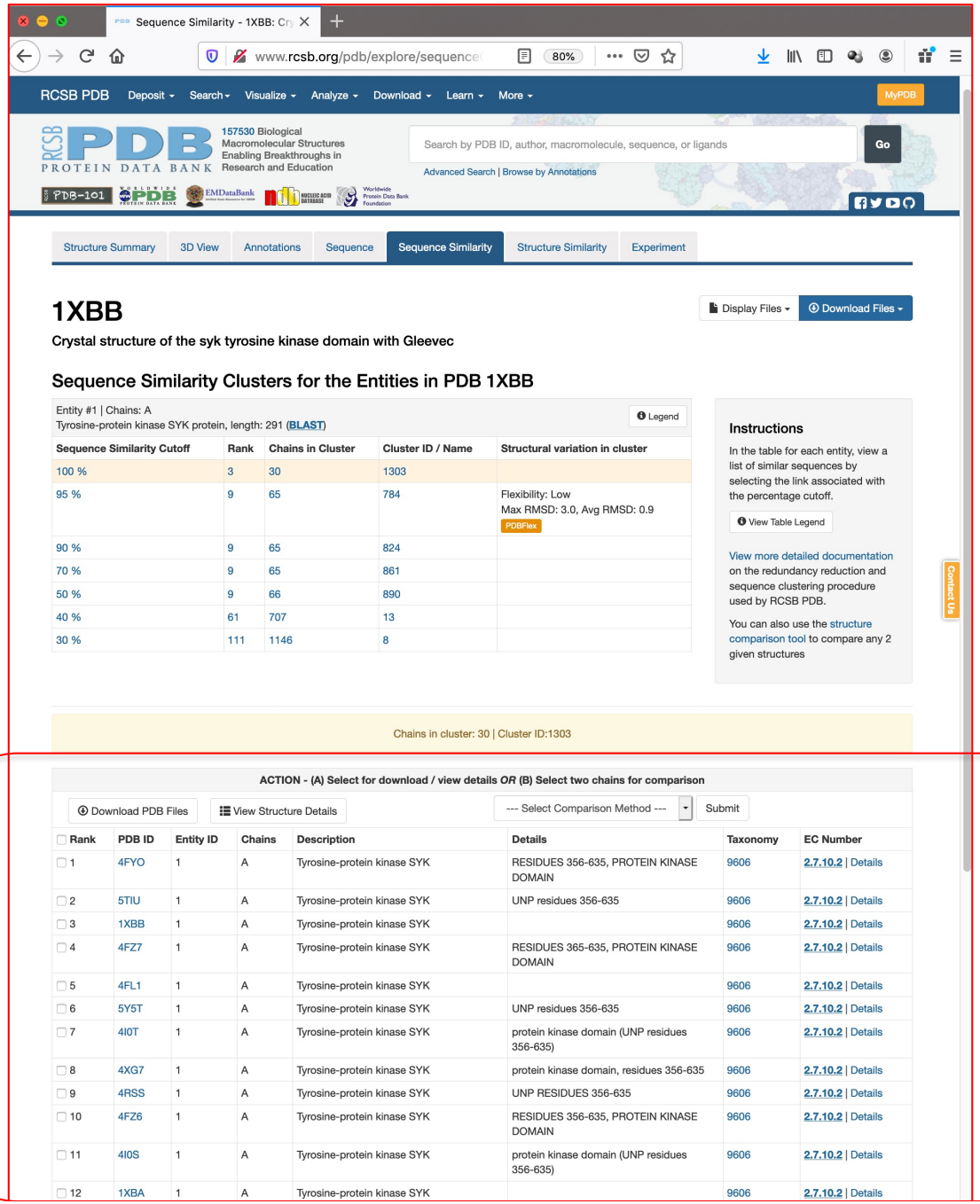
エントリ：1XBBのデータ画面（ページ上部のタブをクリックすると。。。）



構造が決定されているものについて、1xbbに対して配列の類似するものの個数が、類似度のレベルごとに示されている

各行のいずれかの数字をクリック

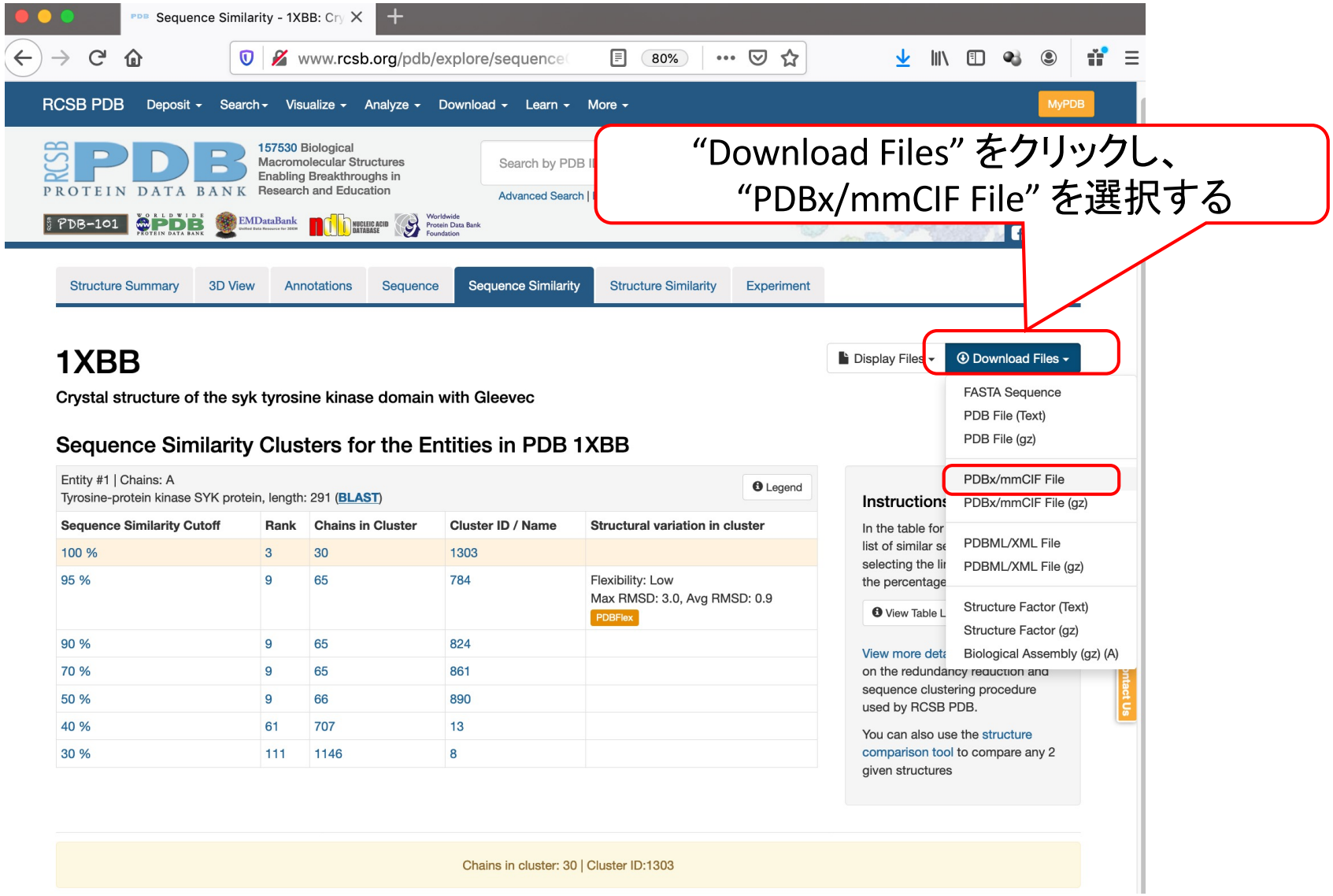
立体構造情報を手に入れる



クリックした行に対応した立体構造の情報が表示される

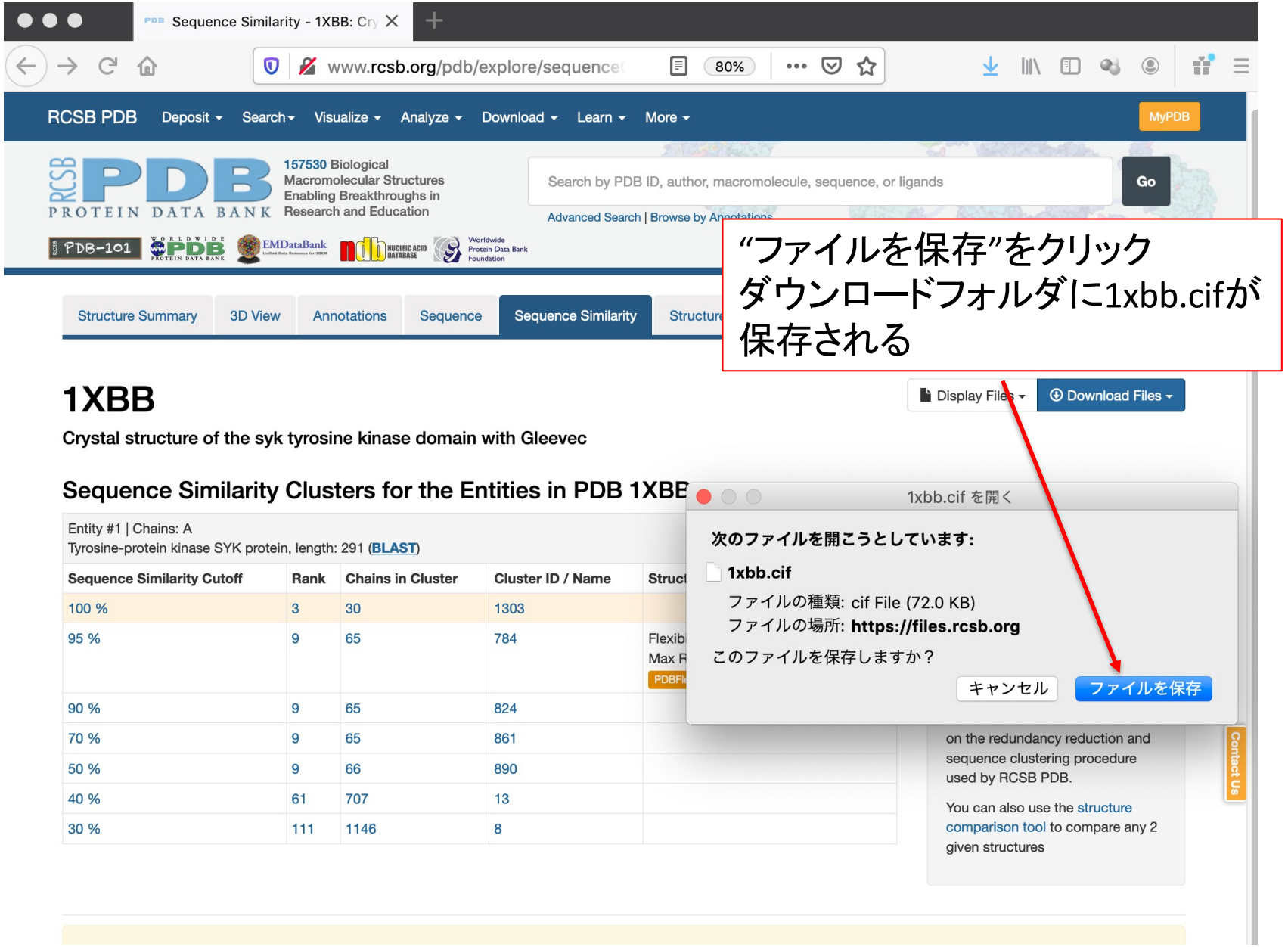
立体構造情報を手に入れる

エントリ：1XBBのデータ画面（トップ）からデータのダウンロード



立体構造情報を手に入れる

エントリ：1XBBのデータ画面（トップ）からデータのダウンロード



座標データを見ていても何も情報が得られない。

立体構造の可視化ツールが必要

分子グラフィクス

PyMol, jViewer, MolMol etc.

今回は、MolMilで分子構造を見る

molmilの使い方　1　－　1個の構造の表示　－

molmil: PDBjで開発されたタンパク質立体構造表示ビューア

ブラウザを立ち上げて、molmil PDBjで検索

Molmil 分子ビューア

このページの他言語版もあります: English

[はじめに] [利用方法] [利用環境] [トラブルシューティング]

はじめに

Molmilはインターネットに接続されたウェブ環境で利用するためにPDBjで開発した新しい分子閲覧ソフトで、できるだけ多くの環境で利用できるように設計されています。OpenGLによるGPUハードウェアアクセラレーションを用い、美しいグラフィックスと最適なパフォーマンスを実現します。 Molmilビューアは既存のサービスに組み込むだけではなく、jVと同じように スタンドアロンビューア としても用いることができます。

スタンドアロンビューアを使って、自身で作成したPDB、PDBML、mmCIF、mmJSONの各フォーマットファイルを読み込むことができるだけではなく、PDB IDやChem Comp IDを指定して既存の構造を読み込むこともできます。 更に、jV用ポリゴンXMLファイルも読み込むことができます。 将来更に機能を追加していく予定です。また、機能追加のリクエストもお待ちしています。 また、Molmilを自分で作ったウェブサイトに埋め込むこともできます。その方法についてはこちらをご覧下さい。

利用方法

Molmilの使い方については下記ページを参照下さい。

- Molmilユーザマニュアル
- 基本的な使い方 (GitHub)
- ウェブページへの埋め込み方 (GitHub)
- 詳細な使い方（Molmil APIの解説） (GitHub)
- Molmil FAQ (GitHub)

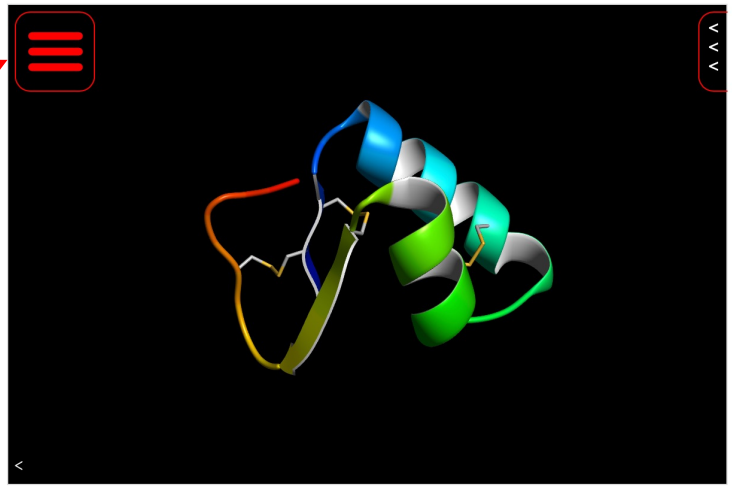
利用環境

Molmilを利用する際、ご利用のブラウザを最新版に更新しておくことをお勧めします。 最新ではないSafariやOperaの中には、WebGLを有効化する必要があるものもあります。 また、ドライバが古すぎない（2010年12月31日以前ではない）ことも確認して下さい。

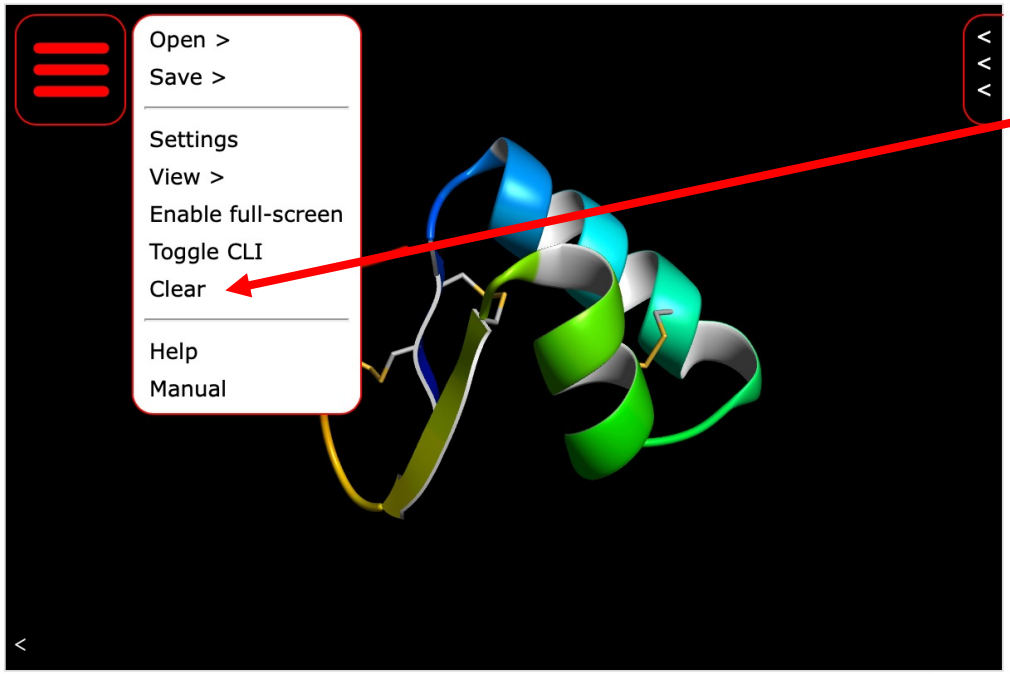
モバイル端末（スマートフォン、タブレット）のうちAndroid、Blackberry場合、Chrome、Firefox、Opera、Blackberryブラウザのいずれかで最新のものをお使い下さい。 iOS（iPhone、iPad）については、現在iOS 8.0は対応しています。

ブラウザがWebGLをサポートしている場合、この下に新しいMolmilビューアを表示することができるでしょう。

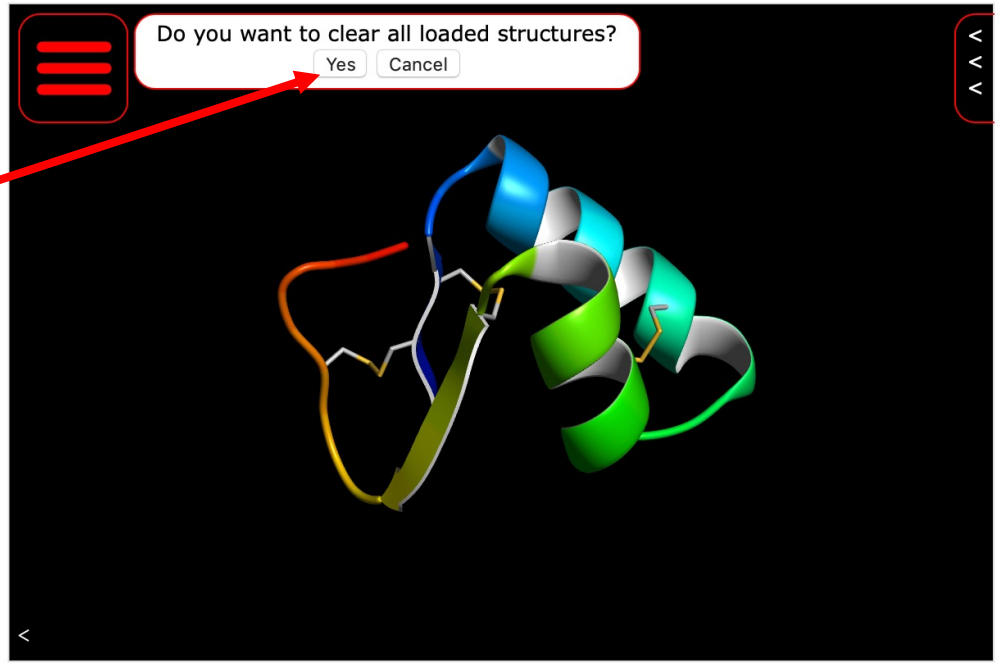
クリック



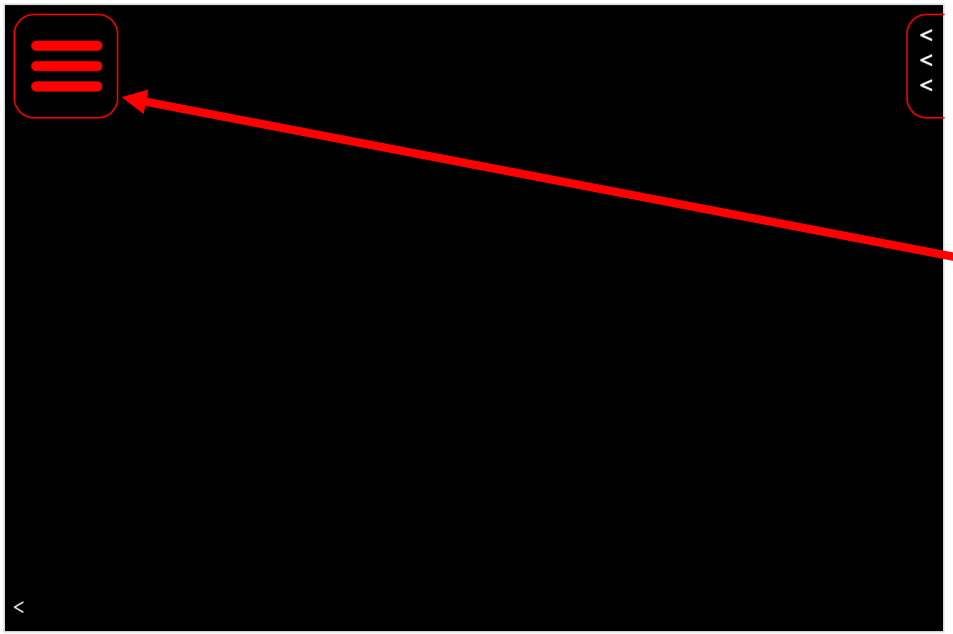
PDBID 1crn: 原子レベルの解像度で解いた疎水性たんぱく質の水構造。クランビン（crambin）の結晶内における水分子の5員環。



Clearをクリック

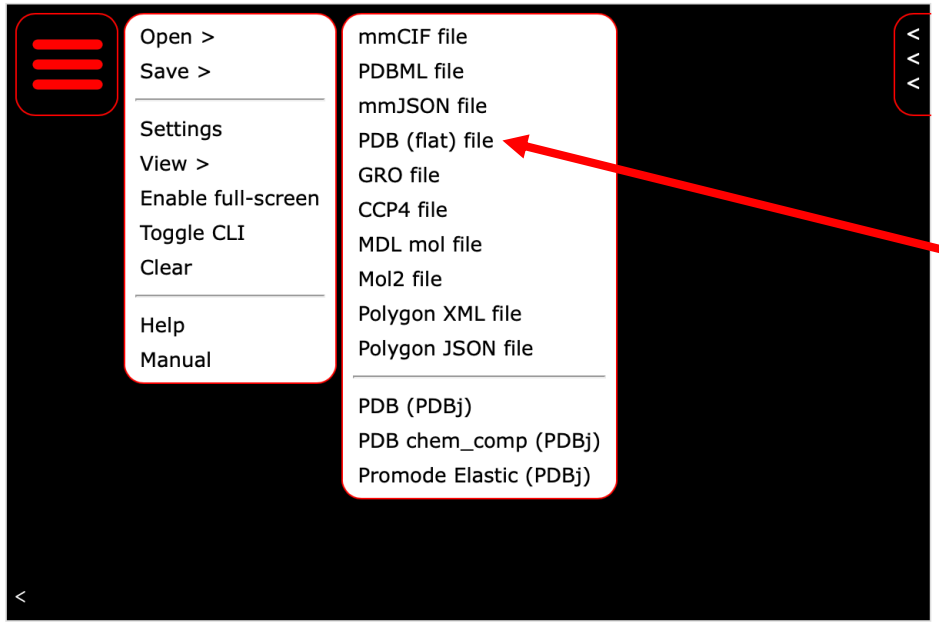


Yesをクリック
デフォルトで表示されている
構造をクリア


クリック

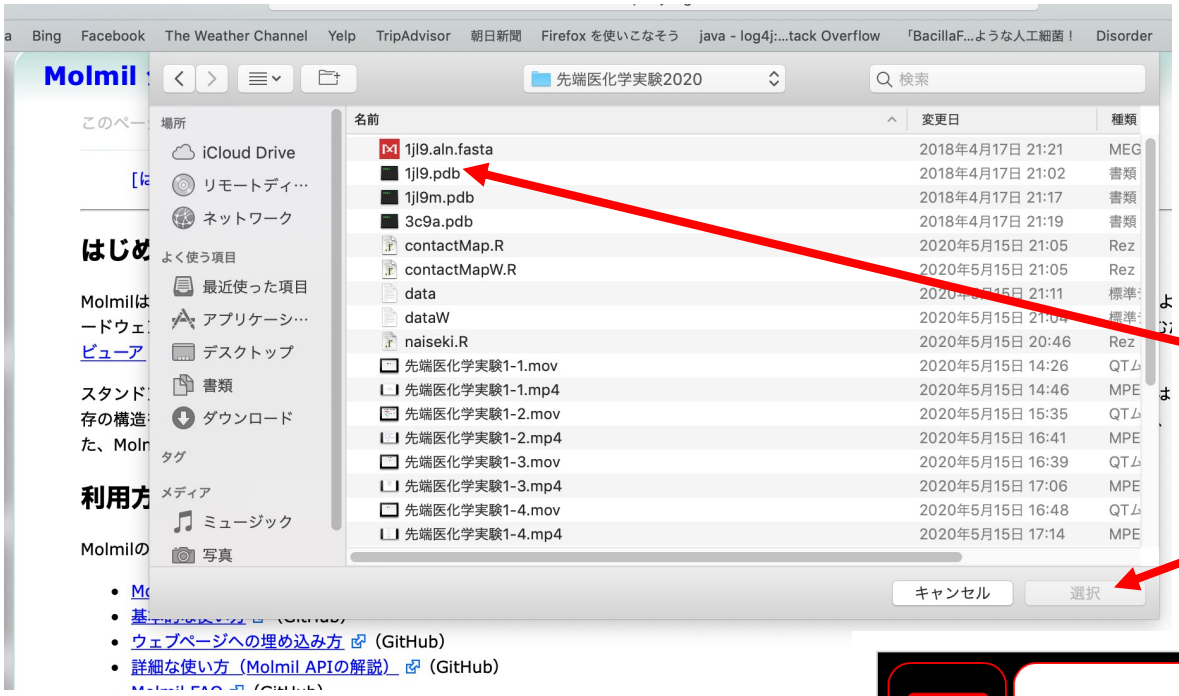
Openをクリック


Open >
Save >
Settings
View >
Enable full-screen
Toggle CLI
Clear
Help
Manual



PDB(flat) file クリック

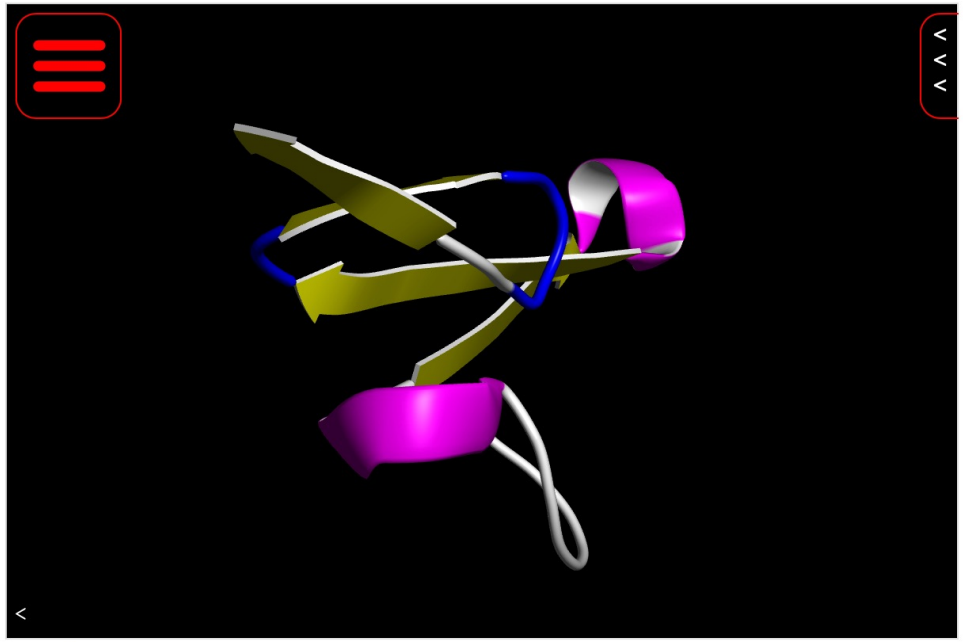
ファイルを選択をクリック



ファイルチューザが開くので
ディレクトリを移動して
1jl9.pdb を選択し、“選択”を
クリック

ファイルが選択されている
ことを確認

Loadをクリック



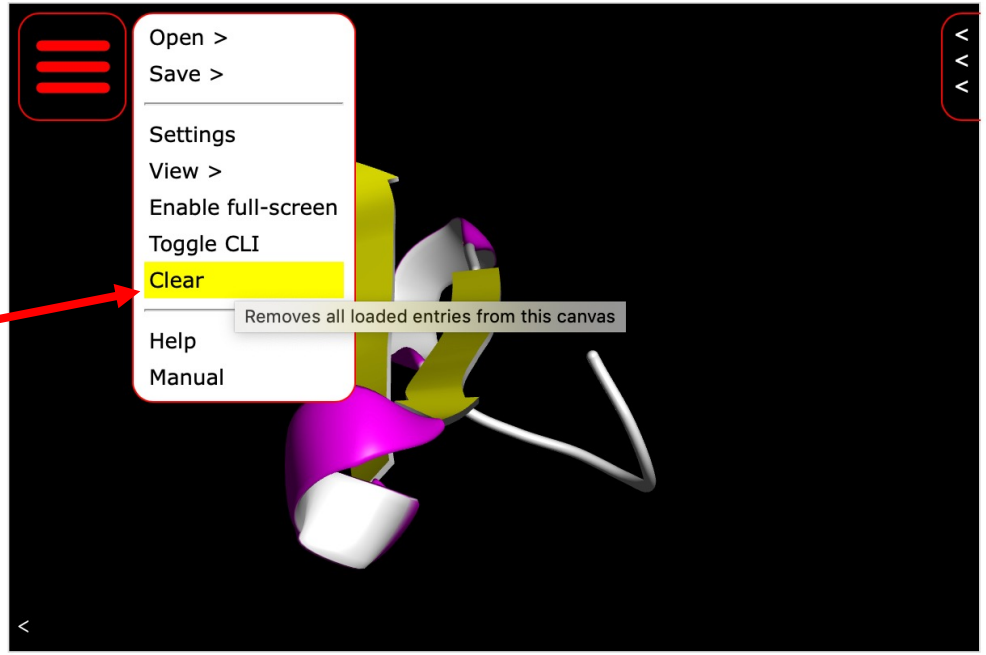
PDBID 1crn: 原子レベルの解像度で解いた疎水性たんぱく質の水構造。クランビン（crambin）の結晶内における水分子の5員環。

1jl9の構造が表示される。

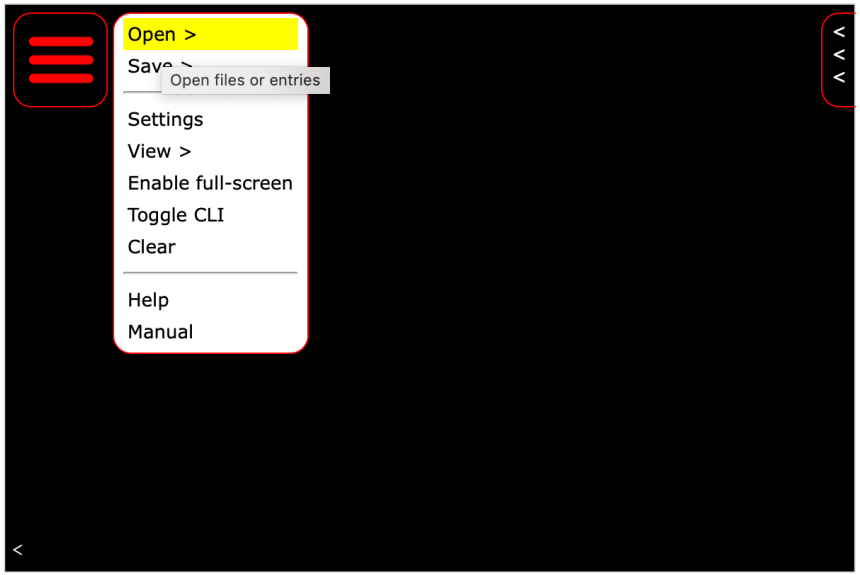
画面を適当にクリックし
クリックしたままドラッグすると
回転することを確認

Shiftキーを押したままで、
クリックしたままドラッグすると
平行移動することを確認

Clearで構造を消す



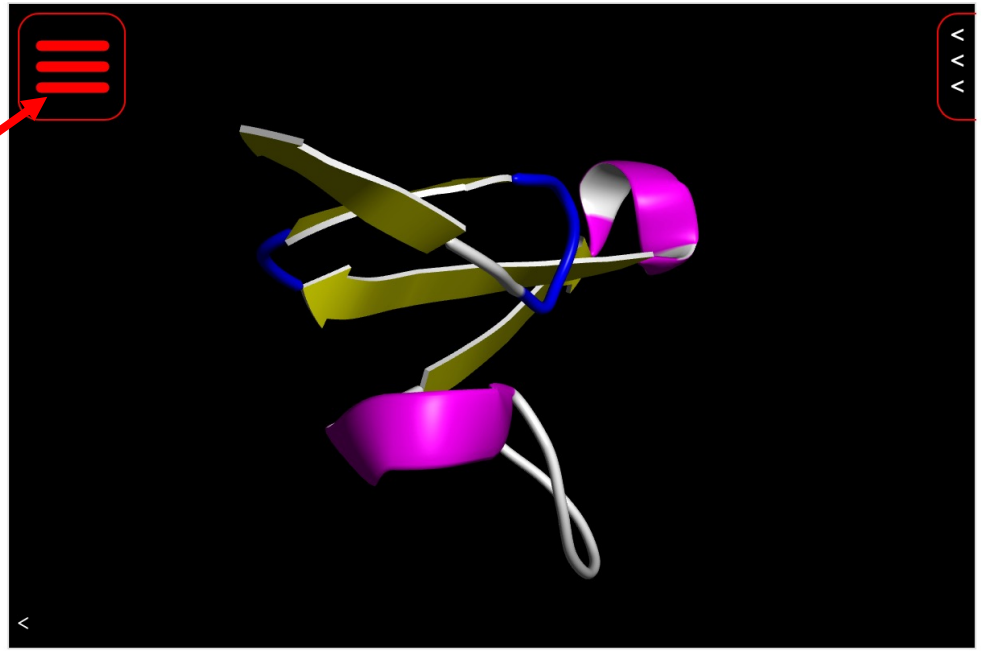
molmilの使い方 2 - 2個の構造の表示 -

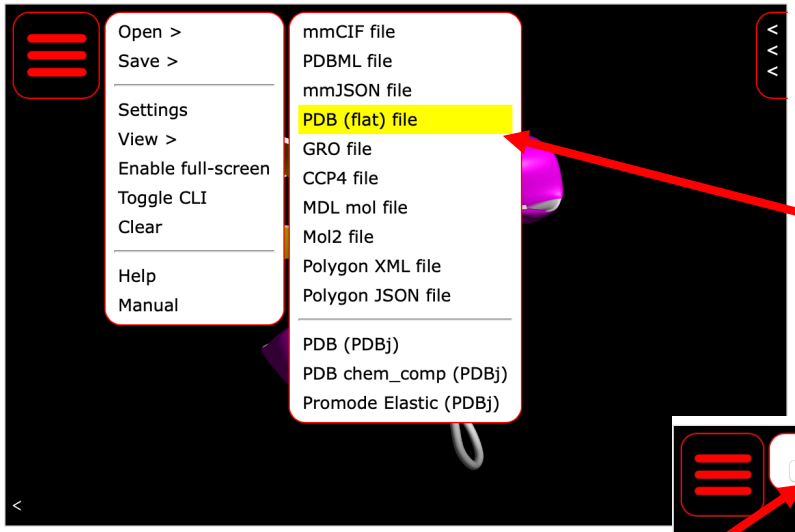


まず、先ほど説明したやり方で
1個の構造(1jl9.pdb)を読み込む

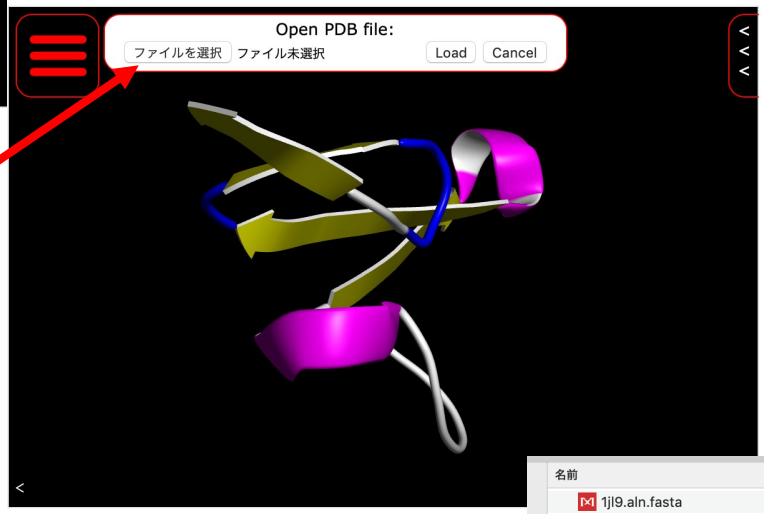
操作は同じなので、読み込み部分は省略

読み込めたらここをクリックして
別の構造おの読み込み操作を行う

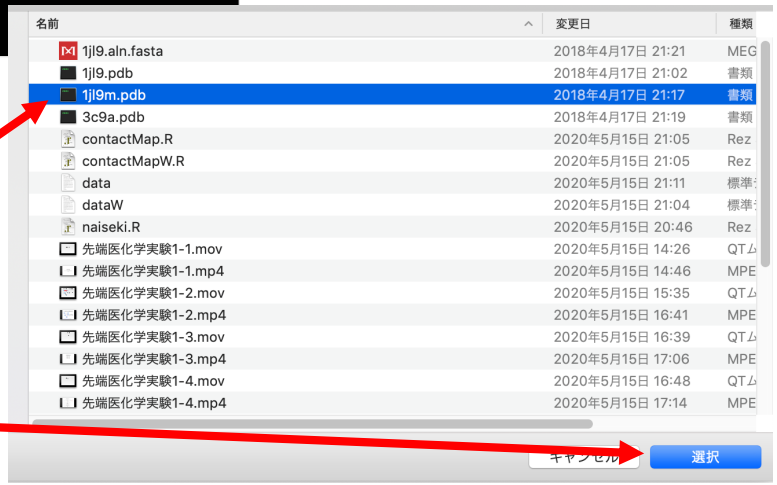




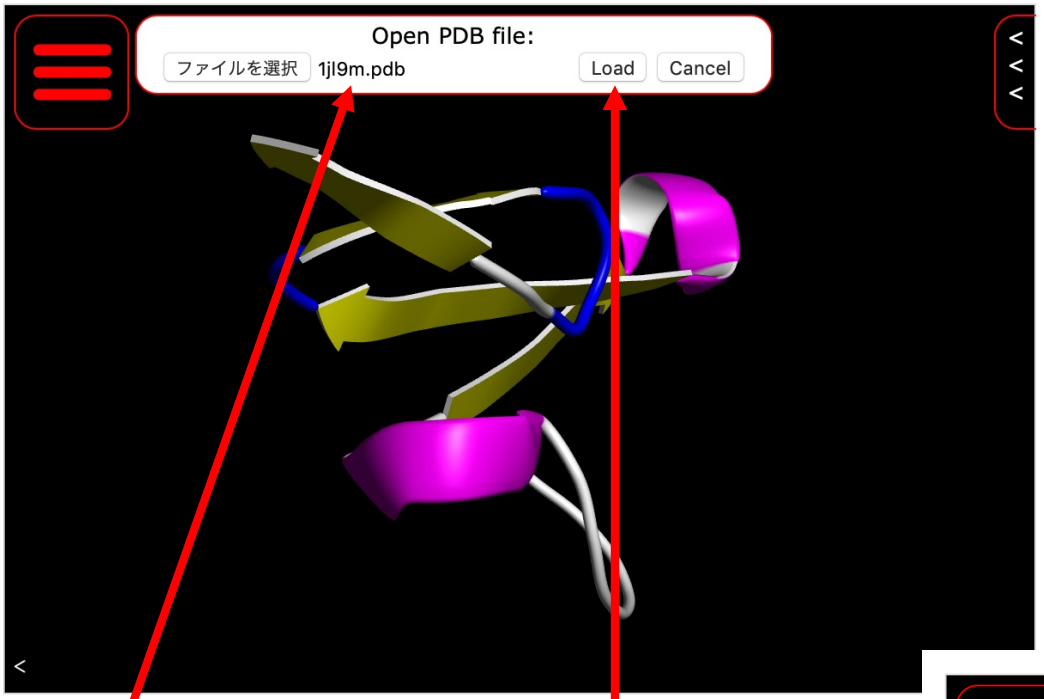
Open -→ PDB (flat) file を選択



ファイルを選択を
クリック



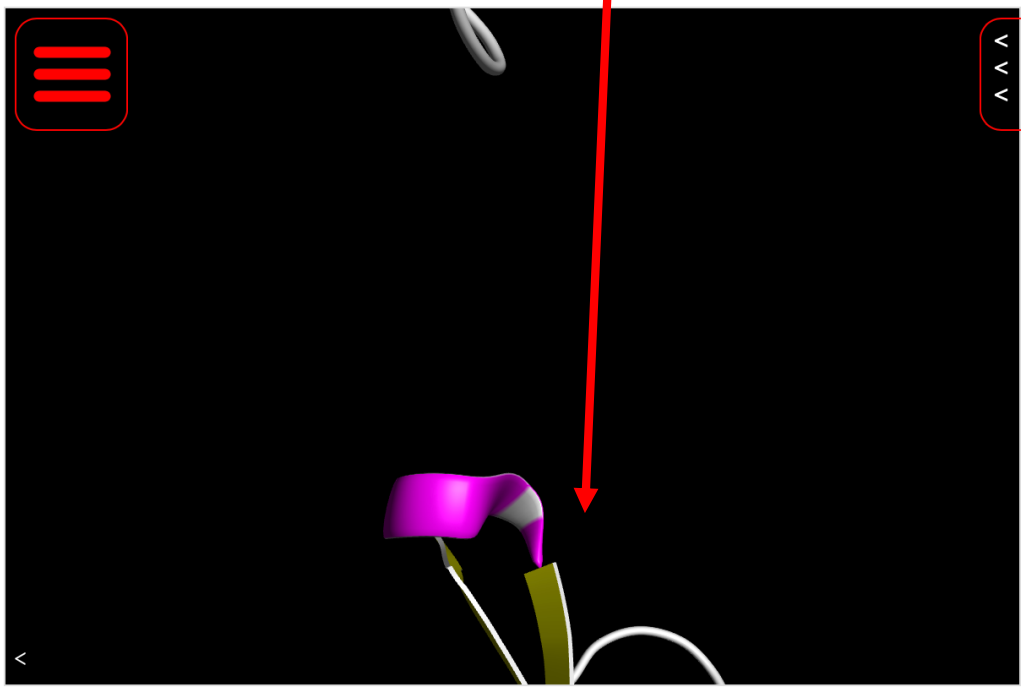
ファイルチューザから
1jl9m.pdbを選択し、
選択をクリック

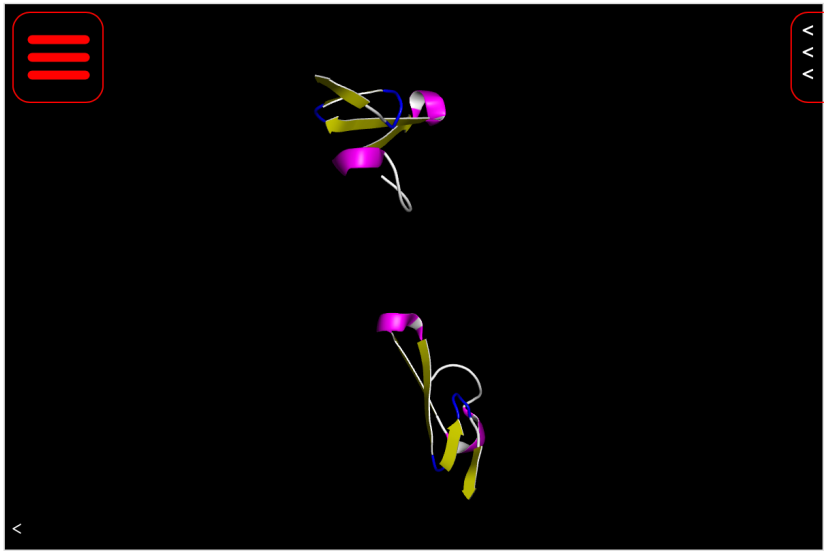


1jl9m.pdbが選択されて
いることを確認

Loadをクリック

2つの構造が表示されているが
スケールがウィンドウサイズに
あわないため全体像が表示され
ていない





マウスのホイールを回転させて
拡大、縮小
あるいはマウスの右ボタンをクリックした
ままドラッグして拡大縮小

平行移動、回転も同様にできる。

1jl9m.pdbは、1jl9.pdbの座標を適当に回転させた後に、平行移動したもの

今回、遺伝的アルゴリズムでこの二つの構造の重ね合わせを行う。
もともと同じ構造なのでピッタリ重なるはず。

1jl9.pdbと遺伝的アルゴリズムで回転させた座標データをmolmilに表示し、本当に
重ね合わせできてるかを確認する

3. Rの復習

今回の実習に使うプログラムはRで書かれている。

プログラムの動作を理解し、またプログラムを書き直すためにRについて復習する

Rとは

統計解析のためにデザインされたプログラミング言語

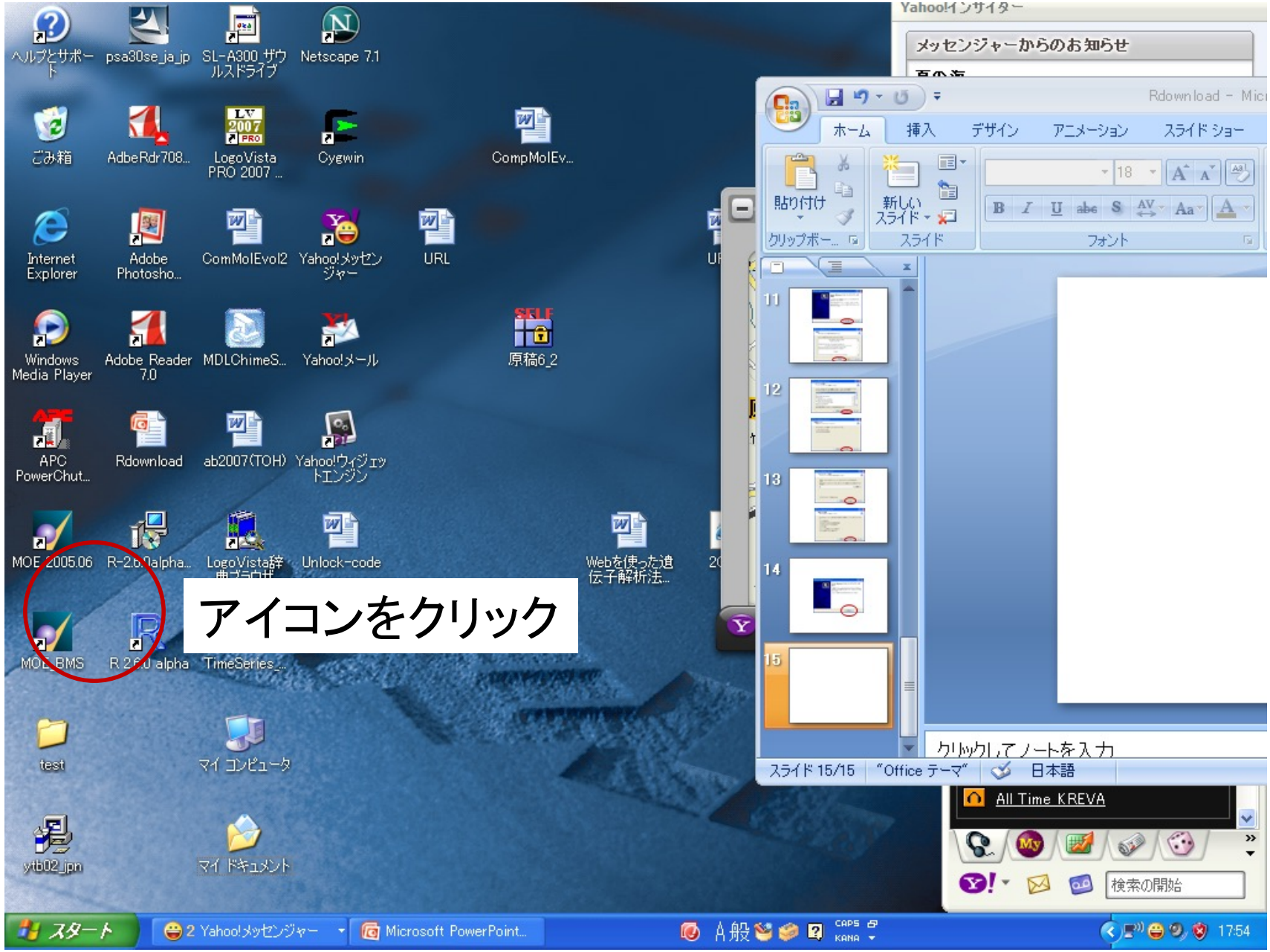
オープンソースのフリーソフトウェア （無料）

マルチプラットフォーム
(Windows, Mac, UNIXなどのOSで同一の作業ができる）

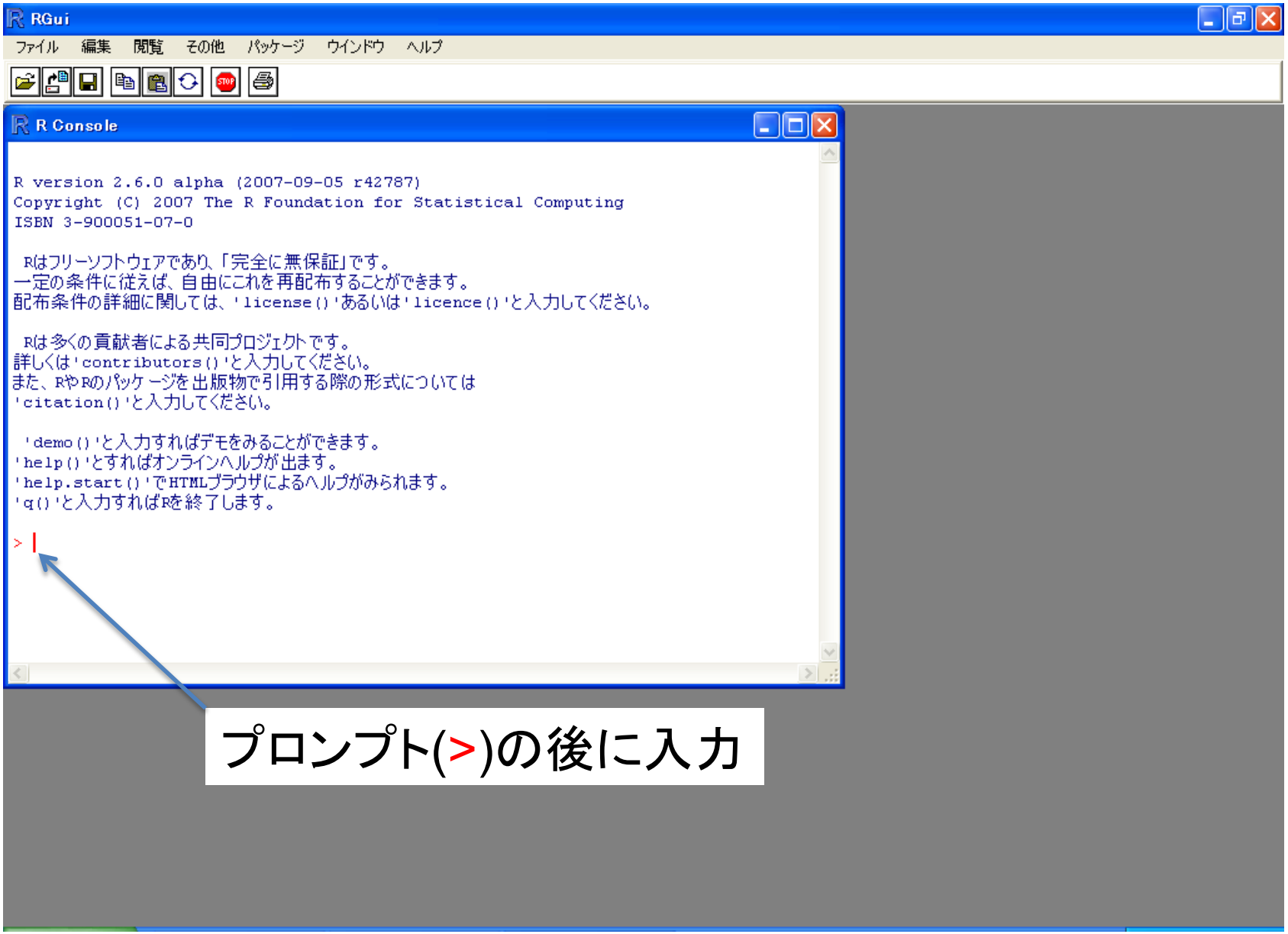
世界各地のRユーザが開発したプログラム（パッケージ）が
CRAN (The Comprehensive R Archive Network)を介して
配信されている。　　(最新の手法が無料で利用可能）

生物系、特にゲノムワイドなデータの解析のパッケージも
充実している　(Bioconductor)

Rの起動



コンソール画面が立ち上がる



プロンプトの後ろにコマンドを入力してEnterを押すと実行

```
> 3 + 4
[1] 7
> 2.3-2.1
[1] 0.2
> 3.14*2.0
[1] 6.28
> 3/4
[1] 0.75
> 1.5^3
[1] 3.375
> 3.2%%3
[1] 0.2
> 9%/%4
[1] 2
```

加

減

乗

除

冪

剰余

切り捨て

変数への値の代入と演算

```
> a <- 2
> a
[1] 2
> print(a)
[1] 2
> b <- 9.7
> b^a
[1] 94.09
> c <- b^a + b/a
> c
[1] 98.94
> b/a
[1] 4.85
```

変数名は、Rの関数名などとして予約
されているものでなければ、半角英数字
を使って自由に指定できる
ただし、変数名を数字ではじめてはいけない
大文字、小文字は区別される

```
> a <- 2
> A <- 3
> print(a)
[1] 2
> print(A)
[1] 3
> ips <- 5
> a.Boss <- 3
> ips <- ips + a.Boss
> ips
[1] 8
```

```
> sqrt(2)
[1] 1.414214
> exp(2)
[1] 7.389056
> exp(1)
[1] 2.718282
> 2.718282^2
[1] 7.389057
> sqrt(exp(2))
[1] 2.718282
```

sqrt　平方根

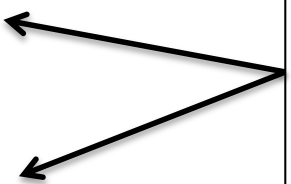
exp　指数関数

関数の引数に関数をいれることもできる

三角関数

sin, cos, tan

```
> sin(pi/2)
[1] 1
> cos(pi/2)
[1] 6.123234e-17
> sin(pi)
[1] 1.224647e-16
> cos(pi)
[1] -1
> tan(0)
[1] 0
> tan(pi/2)
[1] 1.633124e+16
> sin(pi/2)/cos(pi/2)
[1] 1.633124e+16
```



数値的に計算されるので
正確には0であるが、
非常に小さな値として表示

6.123234e-17=6.123234×10^{-17}

本来、無限大になるはず
だが、数値的に計算され
ているので大きな値として
表示

四捨五入

```
> pi
[1] 3.141593
> round(pi)
[1] 3
> round(pi,4)
[1] 3.1416
> round(pi,3)
[1] 3.142
> round(pi,2)
[1] 3.14
```

切り捨て trunc
最大整数 floor
切り上げ ceiling

```
> trunc(-5.678)
[1] -5
> trunc(5.678)
[1] 5
> floor(-5.678)
[1] -6
> floor(5.678)
[1] 5
> ceiling(-5.678)
[1] -5
> ceiling(5.678)
[1] 6
```

Rではベクトルはc()で作成

```
> y <- c(9.02, 20.124, -0.998, 4.787)
> y
[1]   9.020 20.124 -0.998   4.787
```

最大、最小　max, min

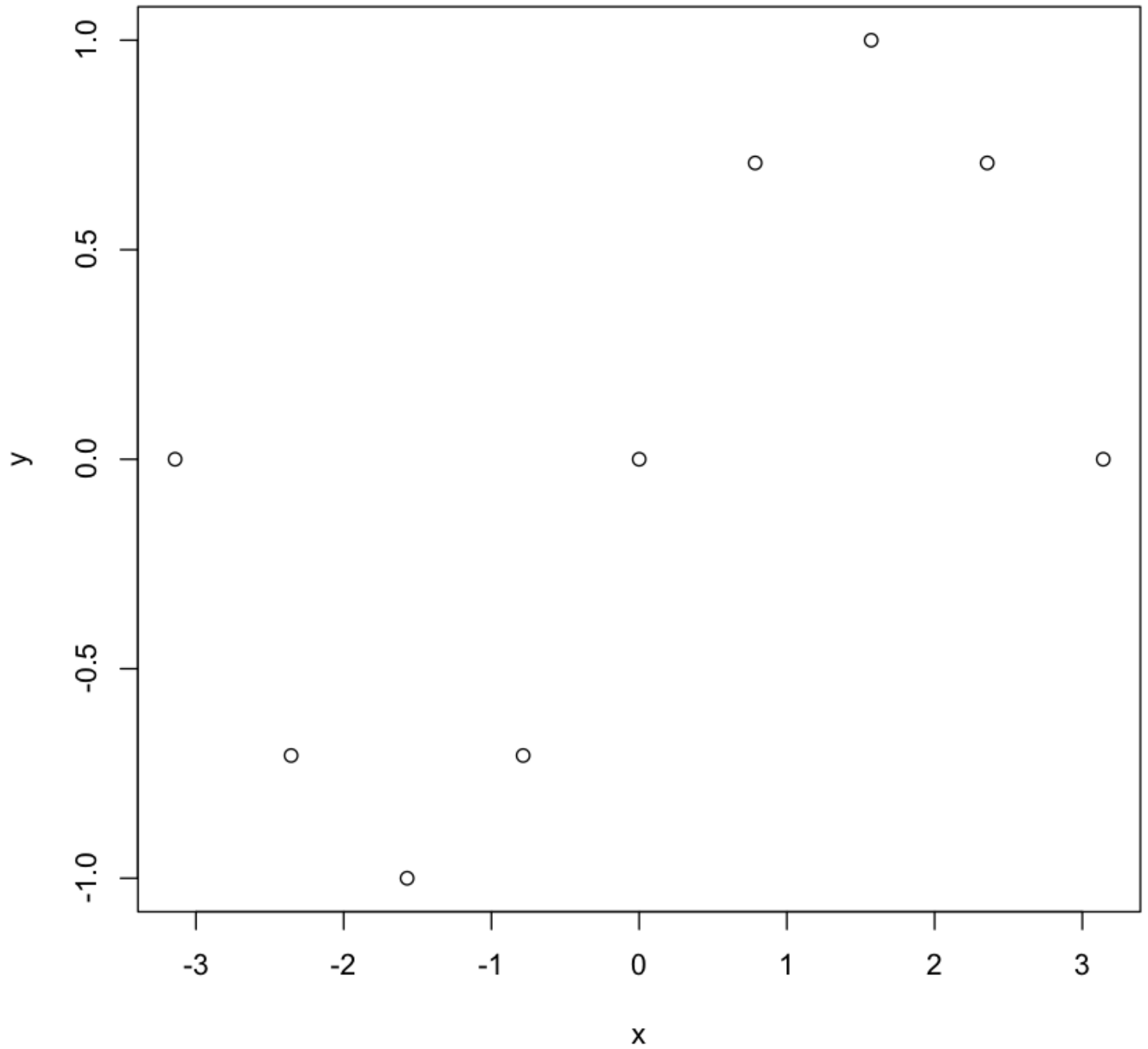
```
> min(y)
[1] -0.998
> max(y)
[1] 20.124
```

```
> x <- c(-pi, -pi*0.75, -pi*0.5, -pi*0.25, 0, pi*0.25, pi*0.5,
pi*0.75, pi)
> y <- sin(x)
> y
[1] -1.224647e-16 -7.071068e-01 -1.000000e+00 -7.071068e-01
0.000000e+00  7.071068e-01  1.000000e+00  7.071068e-01
[9]  1.224647e-16
> plot(x, y)
```

次ページ図

もっと、間隔を細かくとって、より滑らかな曲線を描きたい。
しかし、手作業で x 座標を書くのはめんどうくさい。

次の方法で解決
seq ある区間を、等しい間隔で区切ったベクトルを発生させる


y
1.0
0.5
0.0
-0.5
-1.0
-3
-2
-1
0
1
2
3
x

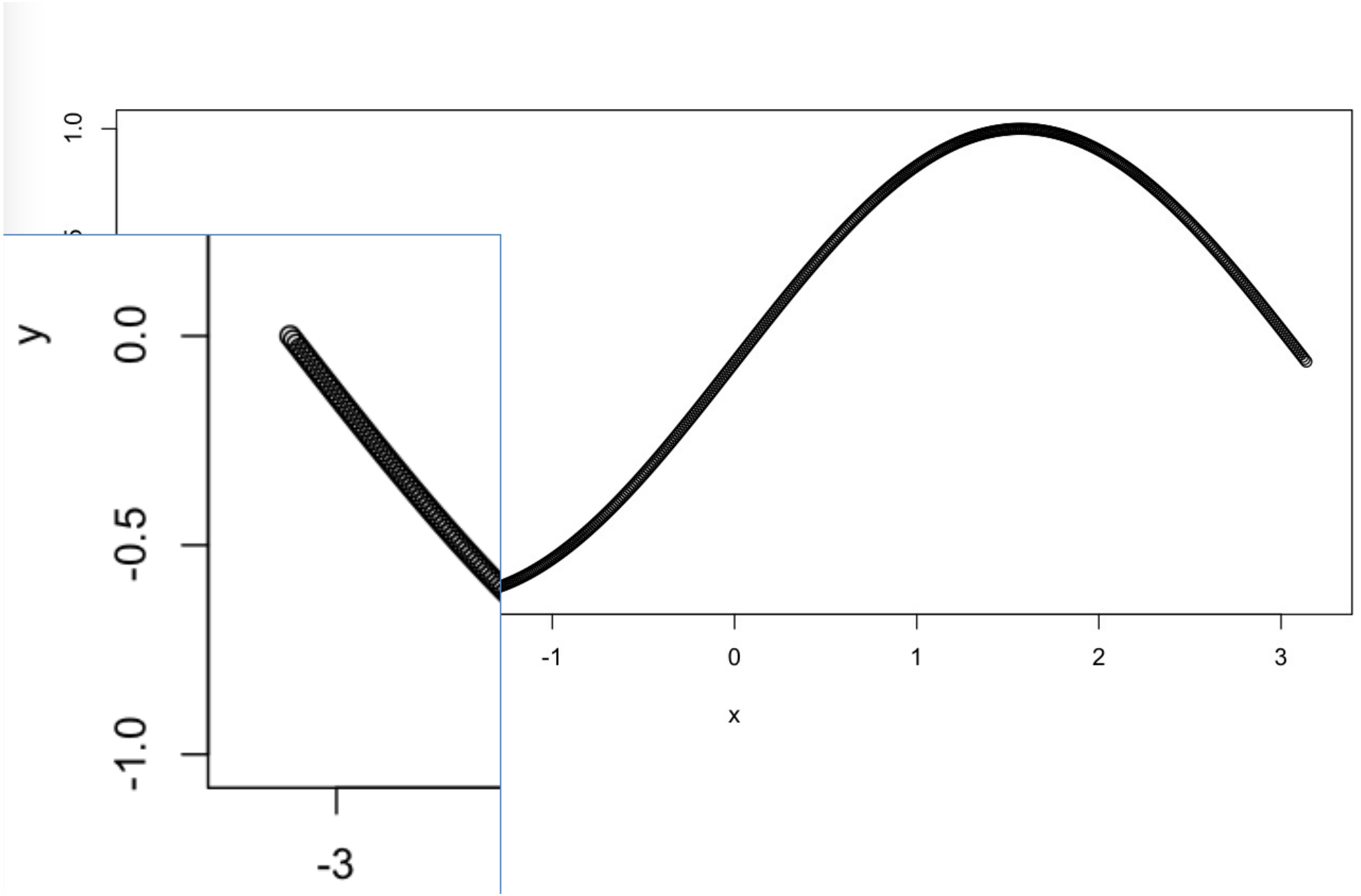
```
> seq(1,5,0.5)
[1] 1.0 1.5 2.0 2.5 3.0 3.5 4.0 4.5 5.0
> seq(3,5,1)
[1] 3 4 5
```

seq(start, end, 間隔)

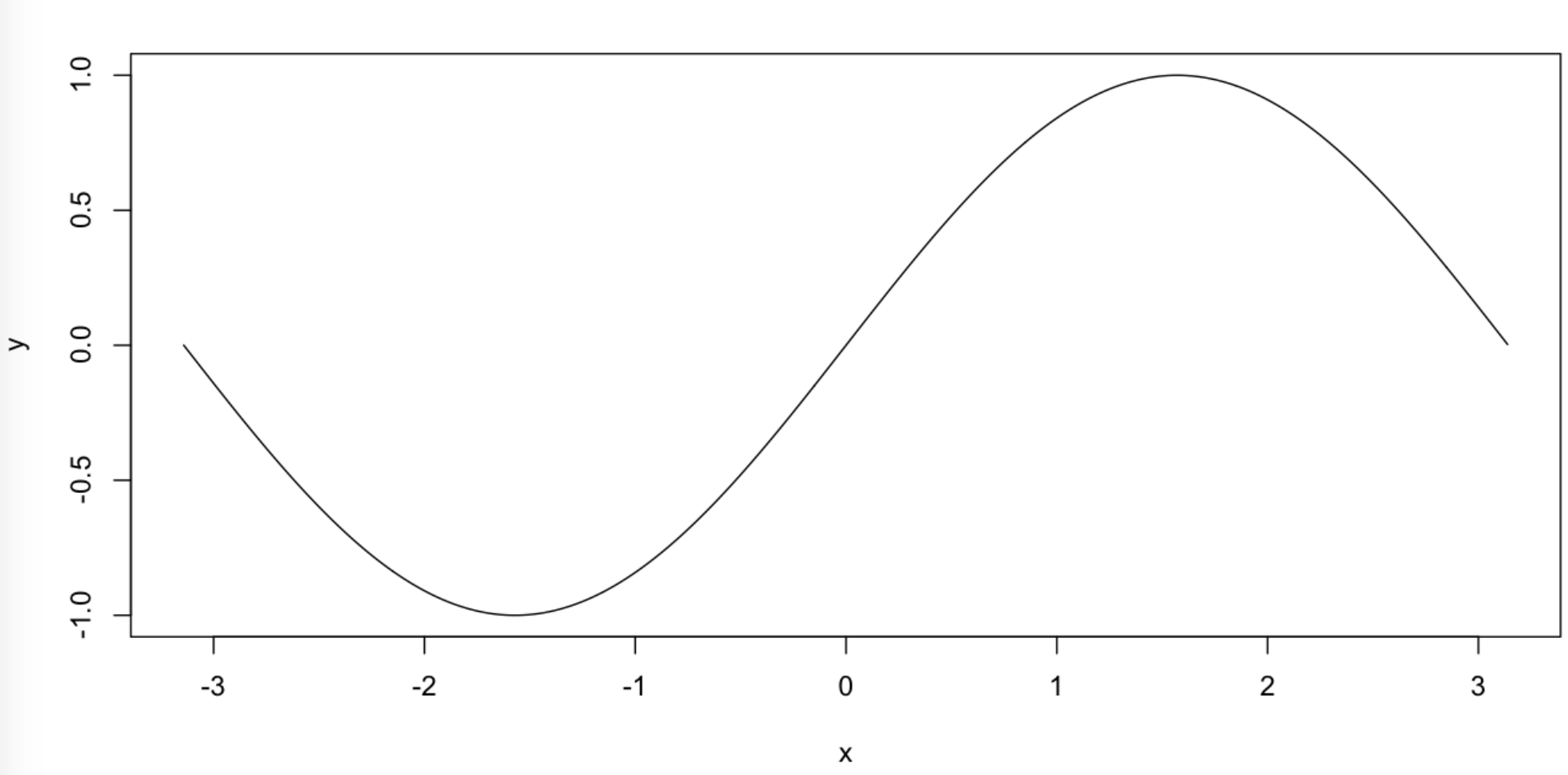
```
> x <- seq(-pi,pi,0.01)
> x
  [1] -3.141592654 -3.131592654 -3.121592654 -3.111592654
…
[628]  3.128407346  3.138407346

> y <- sin(x)
> plot(x,y)
```

図は次ページ

なめらかになったが、ドットで表示するのは見栄えがよくない。
点を打って線でつなぎたい。

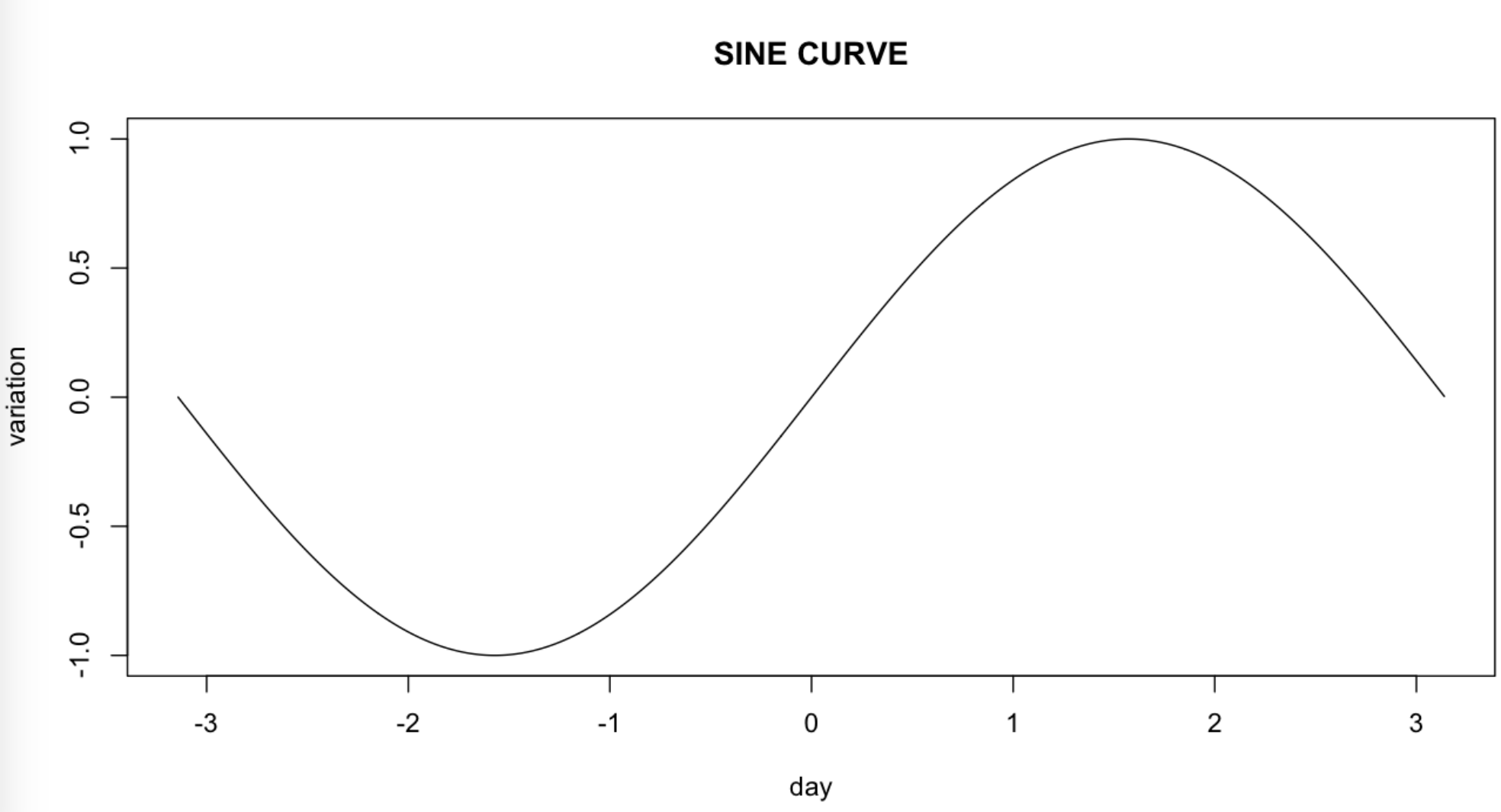

1.0
0.0
-0.5
-1.0
y
-3
-1
0
1
2
3
x

```
> plot(x,y,ty='l')
```



x軸のラベルをday、y軸のラベルをvariation, 全体のタイトルを SINE CURVEとする。

```
> plot(x, y, ty='l',xlab="day", ylab="variation", main="SINE CURVE")
```



生成されたグラフィクスの保存

画像ファイルの形式を、pngあるいはepsで保存したい場合
グラフィクスコマンドを実行する前に
`png(`ファイル名`)`、あるいは`postscript(`ファイル名`)`を実行

グラフィクスコマンド(`plot`や`curve`)を実行した後に

`dev.off()`を実行
Rを実行している作業ディレクトリ(後述)内に、画像ファイル
が作成される。

```
> png("test.png")
> plot(x, y, pch=23)
> dev.off()
quartz
      2
```

スクリーン上にはグラフィクスは表示されない

test.pngをクリックすると図が表示される。

```
getwd()
```
でディレクトリを確認

関数quit()で終了

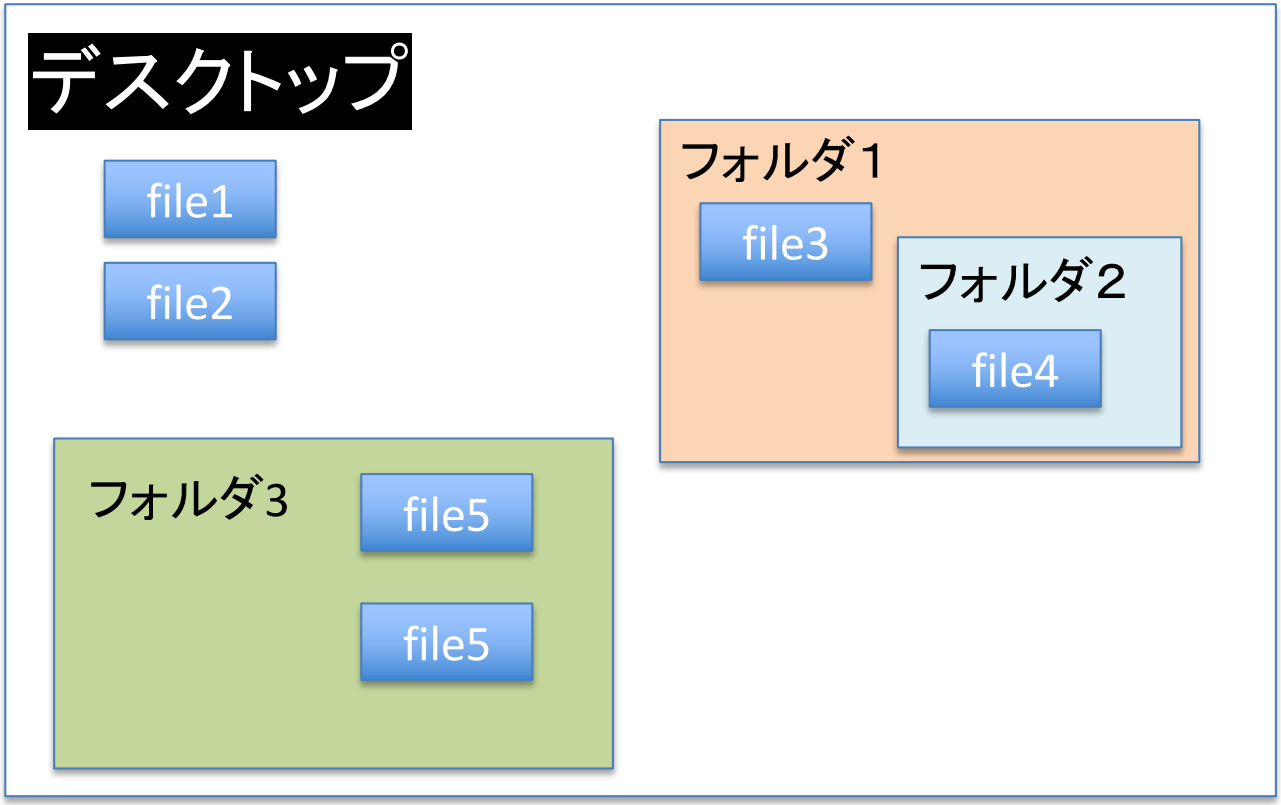

```
> a
[1] 2
> print(a)
[1] 2
> b <- 9.7
> b^a
[1] 94.09
> c <- b^a + b/a
> c
[1] 98.94
> b/a
[1] 4.85
> A <-1
> a
[1] 2
> a.b <-3
> a.b
[1] 3
> 3 -> b
> b
[1] 3
> 4 = c
 以下にエラー 4 = c :  代入の左辺が不正 (do_set) です
> c = 4
> a <- 2
> A <- 3
> print(a)
[1] 2
> print(A)
[1] 3
> ips <- 5
> a.Boss <- 3
> ips <- ips + a.Boss
> ips
[1] 8
> a <- "atgcttgacccgtaat"
> aaseq <- "MLKPAQWELLTGRS"
> a
[1] "atgcttgacccgtaat"
> aaseq
[1] "MLKPAQWELLTGRS"
> aaseq[1]
[1] "MLKPAQWELLTGRS"
> substr(aaseq,2:3)
 以下にエラー substr(aaseq, 2:3) :
   引数 "stop" がありませんし、省略時既定値もありません
> greeting <- "Hello World!"
> print(greeting)
[1] "Hello World!"
> quit()
```

“保存しない”を押して終了してから、再度Rを起動

作業ディレクトリの確認と設定

フォルダ　＝　ディレクトリ

フォルダの中には、ファイルだけでなくフォルダも作れる
※　デスクトップもフォルダの一つ



Rでは、どのフォルダで作業するかを選択できる

getwd() ： 現在のディレクトリ名を出力

list.files() ： 現在のディレクトリ内の
ファイルの一覧を出力
dir(): list.files()と同じ

setwd(“フォルダ名”) ： ワークスペースを二重引用符
で囲まれたフォルダに変更
←-- 作業フォルダを移動するのに使う

ls(): 現在Rが持っているオブジェクト(ベクトルや変数)
の一覧

setwdの使用例

Macの場合

```
setwd("/Users/toh/Desktop")
```

Windowsの場合

```
setwd("c:/mydocuments")
```

今回、作成したRのプログラムやファイルをダウンロードして使用する。
そのプログラムやファイルが置かれているフォルダを作業ディレクトリとすること

ベクトル（vector）

c(),seq以外の方法

```
> 1:50
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
[25] 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48
[49] 49 50
>
> rep(3, 12)                          # 3を12回繰り返す
 [1] 3 3 3 3 3 3 3 3 3 3 3 3
>
> rep(c(5,7), 2)                      # 5と7を2回繰り返す
[1] 5 7 5 7
>
> rep(3:5, 1:3)                       # 3を1回、4を2回、5を3回繰り返す
[1] 3 4 4 5 5 5
>
> x <- 1:50
> x
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41
[42] 42 43 44 45 46 47 48 49 50
> x <- seq(1,50)
> x
 [1]  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24
25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41
[42] 42 43 44 45 46 47 48 49 50
```

```
> x <- c(2,4.5,1.9,1.1,3.3)
> x
[1] 2.0 4.5 1.9 1.1 3.3
> y <- c(4.1,5.5, 1,2.1,2.2)
> y
[1] 4.1 5.5 1.0 2.1 2.2
> length(x)
[1] 5
> length(y)
[1] 5
> z <- 5*x
> z
[1] 10.0 22.5  9.5  5.5 16.5
> x %*% y
      [,1]
[1,] 44.42
```

ベクトルのサイズ

ベクトルの実数倍

ベクトルの内積

```
> x[1]
[1] 2
> y[2]
[1] 5.5
> x[1]*y[1]+x[2]*y[2]+x[3]*y[3]+x[4]*y[4]+x[5]*y[5]
[1] 44.42

> sum(x)
[1] 12.8
> sum(y)
[1] 14.9
```

ベクトルxの要素1を参照

ベクトルyの要素1を参照

```
> x <- matrix(c(1,9,2,8,3,7,4,6), ncol=4)
> x
     [,1] [,2] [,3] [,4]
[1,]    1    2    3    4
[2,]    9    8    7    6
> x <- matrix(c(1,9,2,8,3,7,4,6), ncol=4, byrow=TRUE)
> x
     [,1] [,2] [,3] [,4]
[1,]    1    9    2    8
[2,]    3    7    4    6
```

```
> Z <- matrix(1:6, ncol=3)
>
> dim(Z)      # 行列のサイズ
[1] 2 3
>
> ncol(Z)     # 列の数
[1] 3
>
> nrow(Z)     # 行の数
[1] 2
>
> Z[1,2]      # 要素の取り出し
[1] 3
>
> Z[1,]       # 行の取り出し
[1] 1 3 5
>
> Z[,2]       # 列の取り出し
[1] 3 4
>
> length(Z)   # 全要素の数
[1] 6
```

```
1  3  5
2  4  6
```

繰り返し処理の書き方 (1) for ループ

```
for ( i  in ベクトル )    {
      繰り返し処理の内容
}
```

制御変数 `i` はベクトルの要素の値を毎回とる
(変数名は `i` である必要はない)

ベクトルの次元数だけ処理が繰り返される

通常 ベクトルの与え方は `1:10`
などのようの `i` が1づつ増えるように書くことが多い。

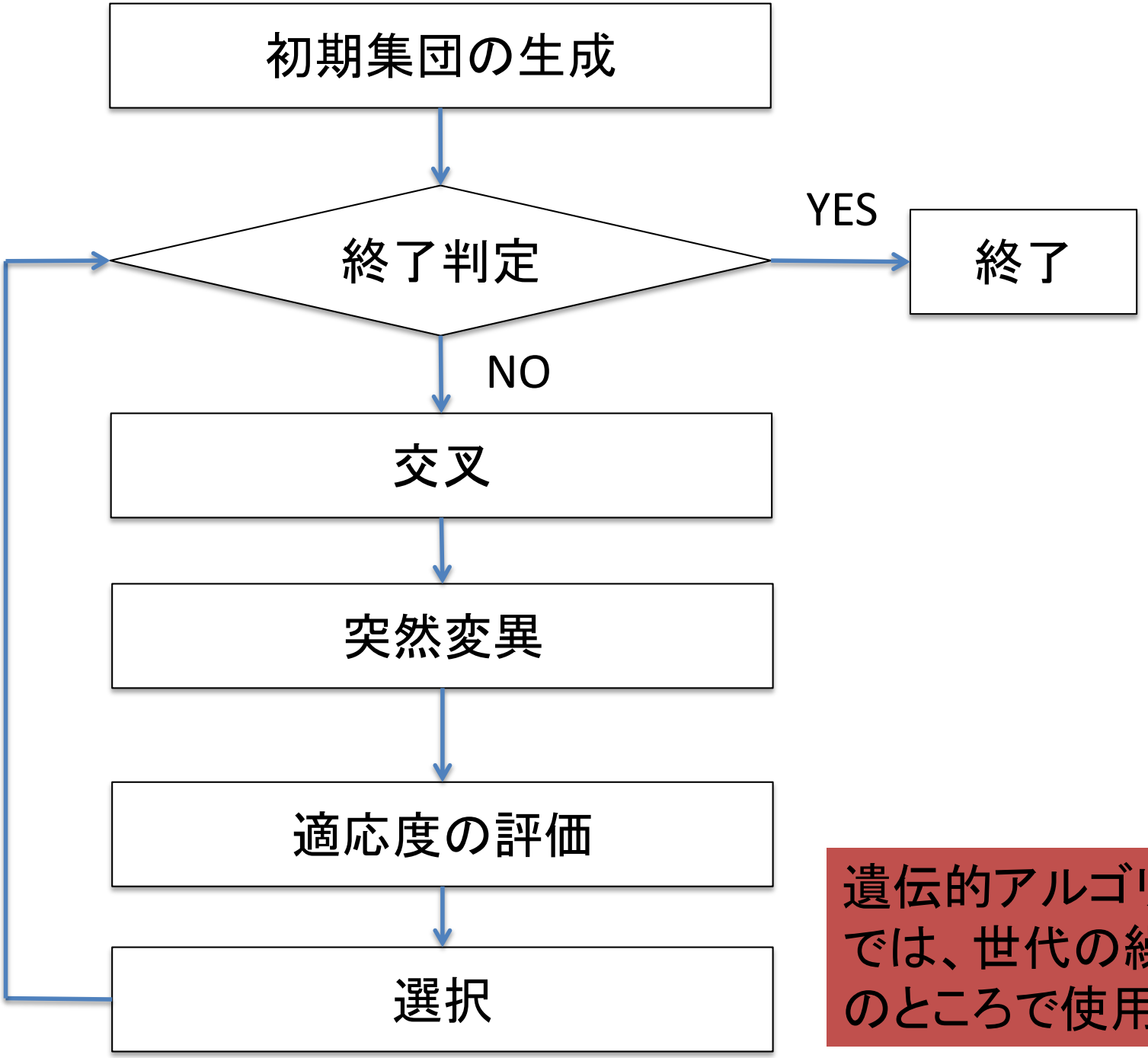
```
j <- 0
for (i in 1:5)  {
  j <- j + i
}
print(j)
```

1から5までの数字を
足す処理

for 文で iは1から5まで
変化し、各i について
{}内の処理が行われる。

i	左辺のj		右辺のj		i
i = 1	1	<-	0	+	1
i = 2	3	<-	1	+	2
i = 3	6	<-	3	+	3
i = 4	10	<-	6	+	4
i = 5	15	<-	10	+	5
	更新されたj		前のステップのj		

R上で 1+2+3+4+5 を実行して同じになるか確認しよう



遺伝的アルゴリズムでは、世代の繰り返しのところで使用

条件分岐

```
if （条件）  {
   条件が真(TRUE)の時の処理
}
else  {
   条件が偽(FALSE)の時の処理
}
```

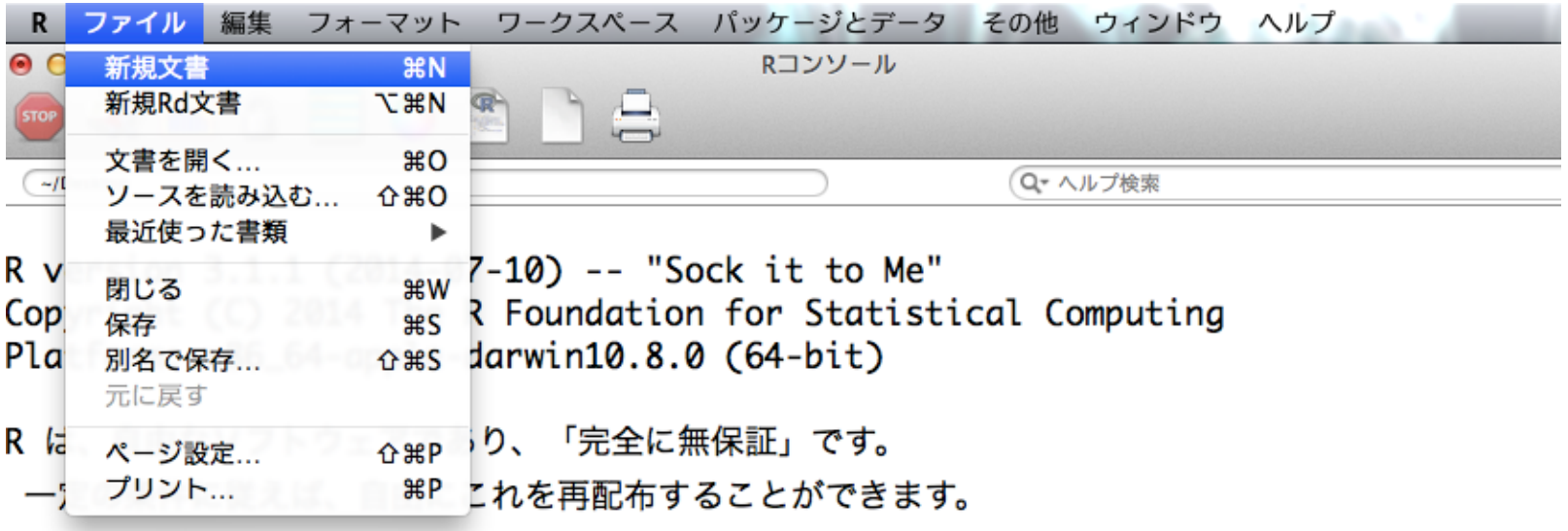
条件の書き方

x == y	一致
x < y	xがyより小
x <= y	xがy以下
x > y	xがyより大
x >= y	xがy以上
x != y	xはyと異なる

== と != は数値だけでなく、文字列についても使える

forループや今回学ぶ条件分岐など、複数行にわたるスクリプトを書く場合、メモ帳やこれから説明するRの エディタに一旦書いて、それをRのコンソールにコピペすると良い。

間違っていても修正が簡単



R v 7-10) -- "Sock it to Me"
Cop R Foundation for Statistical Computing
Pla darwin10.8.0 (64-bit)

R は り、「完全に無保証」です。
これを再配布することができます。
配布条件の詳細に関しては、'license()' あるいは 'licence()' と入力してください。

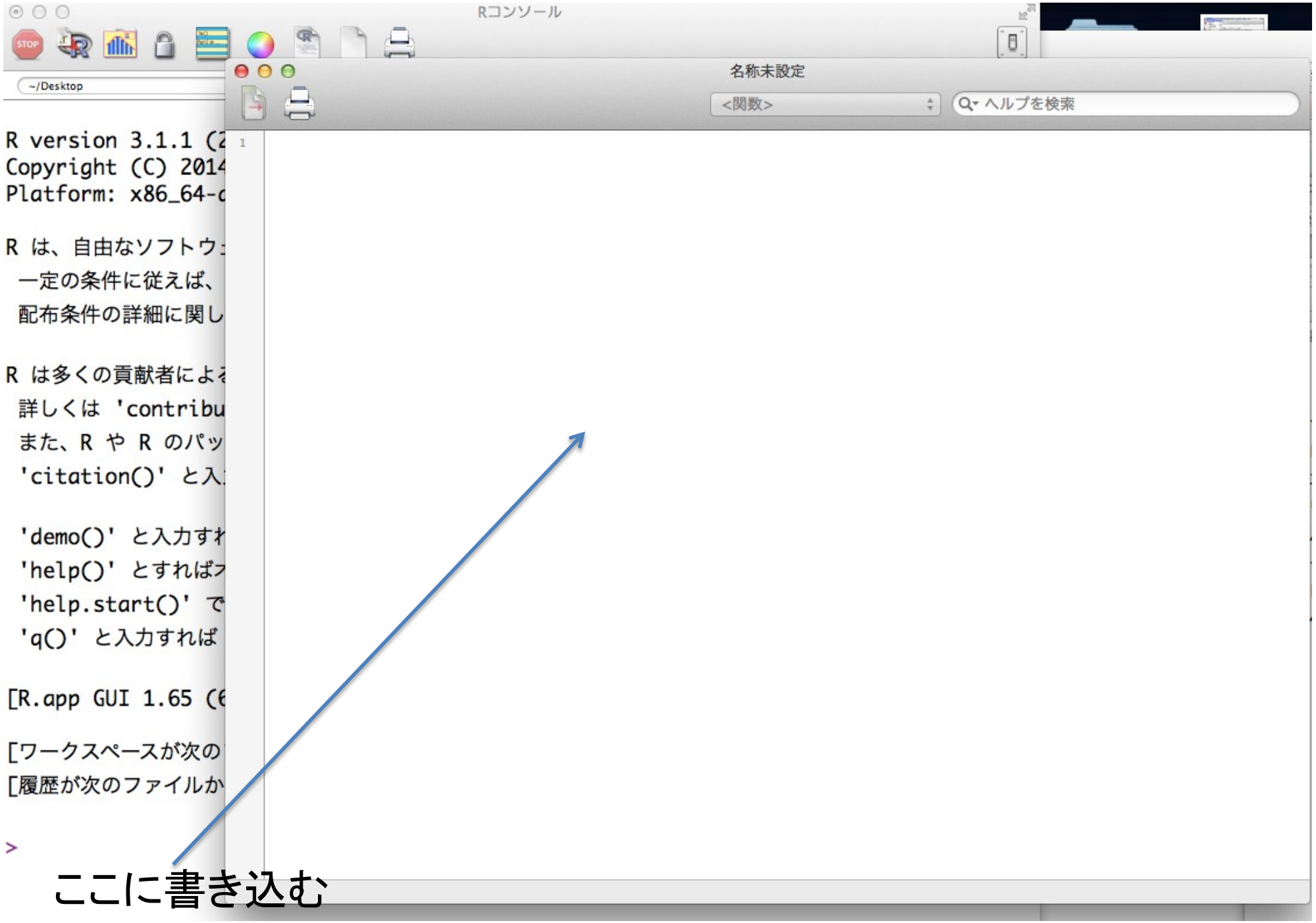
R は多くの貢献者による共同プロジェクトです。
詳しくは 'contributors()' と入力してください。
また、R や R のパッケージを出版物で引用する際の形式については
'citation()' と入力してください。

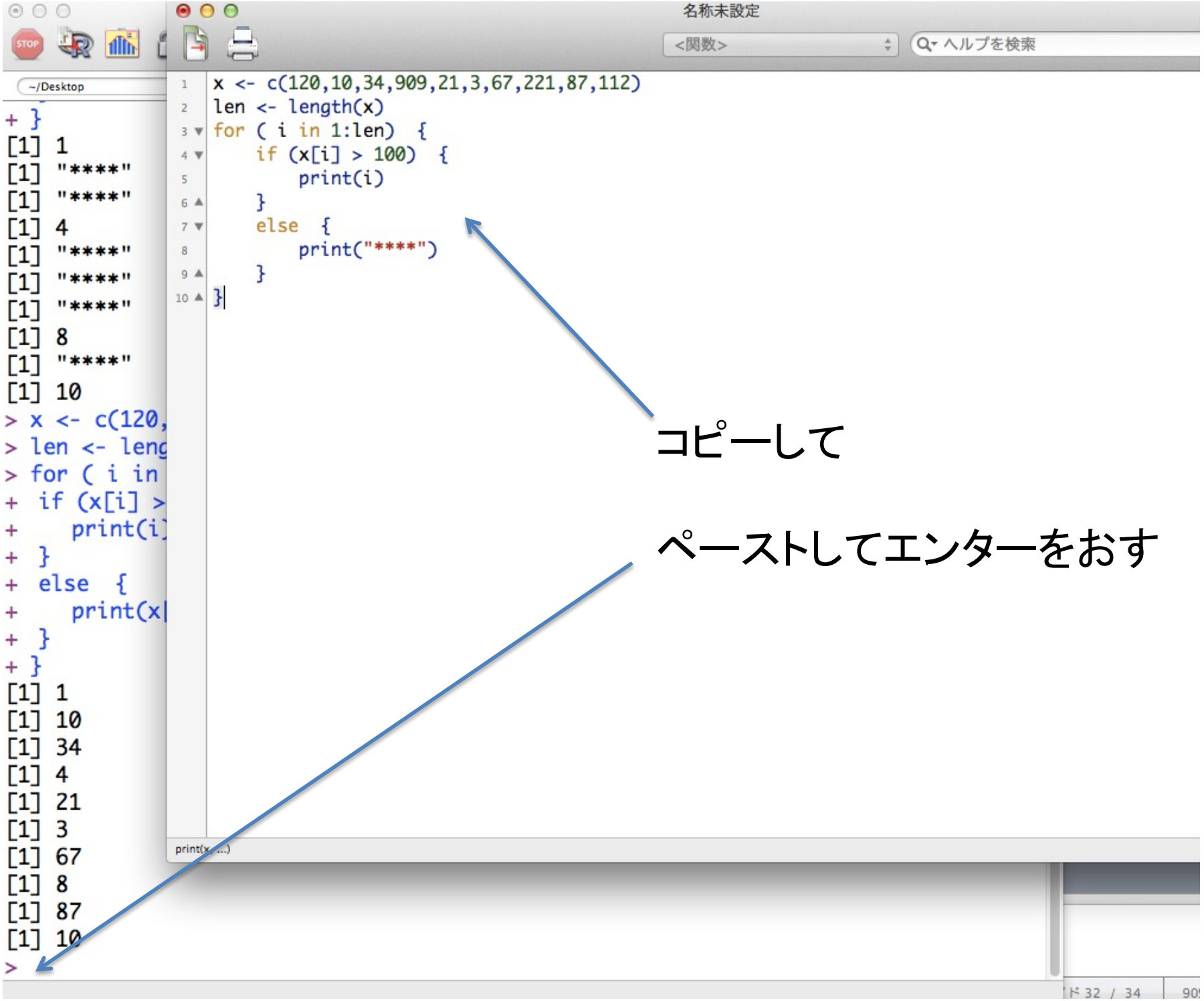
'demo()' と入力すればデモをみることができます。
'help()' とすればオンラインヘルプが出ます。
'help.start()' で HTML ブラウザによるヘルプがみられます。
'q()' と入力すれば R を終了します。

[R.app GUI 1.65 (6784) x86_64-apple-darwin10.8.0]

[ワークスペースが次のファイルから読み込まれました /Users/toh/Desktop/.RData]
[履歴が次のファイルから読み込まれました /Users/toh/Desktop/.Rapp.history]

>


Rコンソール
~/Desktop
R version 3.1.1 (2
Copyright (C) 2014
Platform: x86_64-d
R は、自由なソフトウ
一定の条件に従えば、
配布条件の詳細に関し
R は多くの貢献者によ
詳しくは 'contribu
また、R や R のパッ
'citation()' と入
'demo()' と入力すれ
'help()' とすればヘ
'help.start()' で
'q()' と入力すれば
[R.app GUI 1.65 (6
[ワークスペースが次の
[履歴が次のファイルか
>
名称未設定
<関数>
ヘルプを検索
1
ここに書き込む


名称未設定
<関数>
ヘルプを検索
~/Desktop
x <- c(120,10,34,909,21,3,67,221,87,112)
len <- length(x)
for (i in 1:len) {
if (x[i] > 100) {
print(i)
}
else {
print("****")
}
}
+ }
[1] 1
[1] "****"
[1] "****"
[1] 4
[1] "****"
[1] "****"
[1] "****"
[1] 8
[1] "****"
[1] 10
> x <- c(120,
> len <- leng
> for (i in
+ if (x[i] >
+ print(i
+ }
+ else {
+ print(x
+ }
+ }
[1] 1
[1] 10
[1] 34
[1] 4
[1] 21
[1] 3
[1] 67
[1] 8
[1] 87
[1] 10
>
コピーして
ペーストしてエンターをおす

```
x <- c(120,10,34,909,21,3,67,221,87,112)
len <- length(x)
for ( i in 1:len)  {
   if (x[i] > 100)  {
      print(i)
   }
   else  {
      print("****")
   }
}
```

ベクトル xの要素が100より大なら
　i を出力
さもなければ
　**** を出力

```
[1] 1
[1] "****"
[1] "****"
[1] 4
[1] "****"
[1] "****"
[1] "****"
[1] 8
[1] "****"
[1] 10
>
```

複数の条件を組み合わせる

条件1　&&　条件2
　条件1も条件2の真であれば真
　その他は、偽

条件１　||　条件2
　条件1か条件2のいずれかが真であれば真
　両方が偽の時に偽

２条件だけでなく
(条件1 && 条件2) || 条件3
などのように複数の条件を組み合わせて利用できる

問題1

ベクトルの要素が偶数である以下、10以上70以下の時だけ、そのxの要素を書き出すコード

```
x <- c(120,10,34,909,21,3,67,221,87,112)
len <- length(x)

for (i in 1:len)  {
      if (                                    ) {
      print(x[i])
   }
   else  {
      print(i)
   }
   }
}
```

x[i] は偶数か、x[i]が10以上70以下という条件はどうなるか？

偶数であること => 2で割った余りが0

Rには様々な関数が容易されているが自分の
データの処理には足りない場合がある。

自分のオリジナル関数を作成しよう

Rでは、自分の関数は次のように作成する。

関数名 <- function(引数1, 引数2, …)　{

　　括弧内の引数を使った処理を記述

　　return (戻り値)
}

例えば xとyの積を求める関数は次のように書ける。

```
xy <- function(x, y)  {
    return (x*y)
}
```

※ 引数をx, yにする必然性はなく、他の変数名でも構わない
処理の記述部分には引数として与えられた変数名で記述

プログラムのファイルは拡張子をRとしておく。
ドキュメントフォルダ（Zドライブ）におく。

xy.Rをsourceで、実行中のコンソール画面から読み込むと
オリジナル関数xyを利用できるようになる。

```
> source("Z:xy.R")
> xy(4,5)
[1] 20
> xy(1.41421356, 1.41421356)
[1] 2
```

問題2

forループを使って、与えられた2つのベクトルの内積をもとめるプログラムを作成してみる。

naiseki.Rを穴埋めする形で作成

```
naiseki <- function(a, b)  {
  len <- [          ]
  j <- 0
  for (i in 1:len)  {
    j <- j + [          ]
  }
  return (j)
}
```

データフレーム

行列は1種類の型のみで構成される表形式のデータ

```
> x <- matrix(c(1,5,9,2),ncol=2)
> typeof(x)
[1] "double”
> x <- matrix(c("a","b","c","d","ef","gde"), ncol=3)
> typeof(x)
[1] "character”
> x
     [,1] [,2] [,3]
[1,] "a"  "c"  "ef"
[2,] "b"  "d"  "gde"
```

行列で数字と文字を混在させると、自動的に数字は文字型にされる

```
> x <- matrix(c("a","b","c","d",1,2), ncol=3)
> x
     [,1] [,2] [,3]
[1,] "a"  "c"  "1"
[2,] "b"  "d"  "2"
> typeof(x)
[1] "character"
```

行列では文字型として扱われる

行列は1種類のデータ型のみで構成される!!

------→ 複数のデータ型を混在させることのできる表形式のデータをRではデータフレームとよぶ

データフレーム作成　1

(1) ベクトルとして作成したデータをつなげる

```
Ht <- c(165.6, 163.3, 172.4, 175.3)
Sx <- c("male", "female", "female", "male")
a <- data.frame(height=Ht, Sex=Sx)

a
  height    Sex
1  165.6   male
2  163.3 female
3  172.4 female
4  175.3   male
```

データフレーム作成 2

(2) ファイルからの読み込み

```
x <- read.table("data.txt")
x
 V1   V2      V3
1   1.95 3.3    RED
2   1.55 2.1 YELLOW
3   1.45 2.2 YELLOW
4   2.01 3.4    RED
5   2.13 3.0    RED
6   1.54 1.9 YELLOW
7   1.98 2.9    RED
8   1.47 1.8 YELLOW
9   1.34 1.7 YELLOW
10 1.36 1.8 YELLOW
11 2.23 3.1    RED
```

データフレーム作成　3

(2) データフレームの各列に名前をつける

```
names(x) <- c("activity","size","color")
x
   activity size  color
1      1.95  3.3    RED
2      1.55  2.1 YELLOW
3      1.45  2.2 YELLOW
4      2.01  3.4    RED
5      2.13  3.0    RED
6      1.54  1.9 YELLOW
7      1.98  2.9    RED
8      1.47  1.8 YELLOW
9      1.34  1.7 YELLOW
10     1.36  1.8 YELLOW
11     2.23  3.1    RED

names(x)
[1] "activity" "size"     "color"
```

データフレーム x

1.95	3.3	RED
1.55	2.1	YELLOW
1.45	2.2	YELLOW
2.01	3.4	RED
2.13	3.0	RED
1.54	1.9	YELLOW
1.98	2.9	RED
1.47	1.8	YELLOW
1.34	1.7	YELLOW
1.36	1.8	YELLOW
2.23	3.1	RED

x[,1]　　x[,2]　　x[,3]

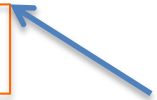
それぞれ、xの1列、2列、3列をベクトルとして抽出

データフレーム x

1.95	3.3	RED
1.55	2.1	YELLOW
1.45	2.2	YELLOW
2.01	3.4	RED
2.13	3.0	RED
1.54	1.9	YELLOW
1.98	2.9	RED
1.47	1.8	YELLOW
1.34	1.7	YELLOW
1.36	1.8	YELLOW
2.23	3.1	RED

x[1,]

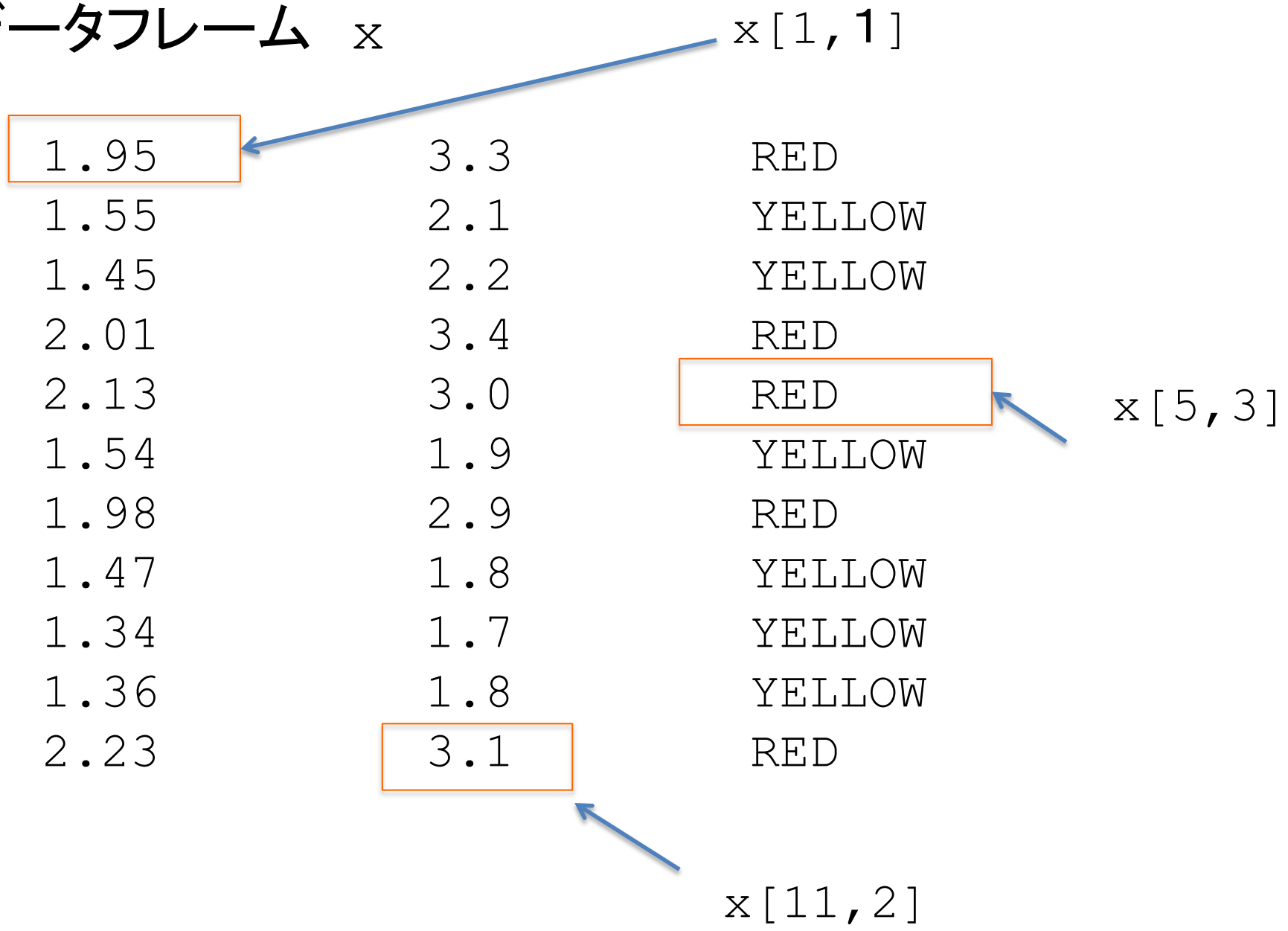
x[5,]



x[11,]

xの1行、5行、11行をベクトルとして抽出

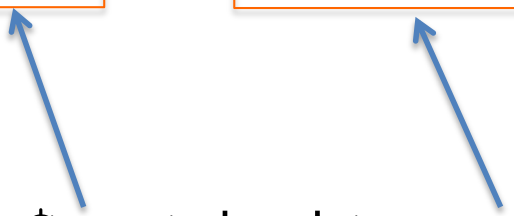
データフレーム x



添字でxの要素を添字を指定して抽出

データフレーム　x

1.95	3.3	RED
1.55	2.1	YELLOW
1.45	2.2	YELLOW
2.01	3.4	RED
2.13	3.0	RED
1.54	1.9	YELLOW
1.98	2.9	RED
1.47	1.8	YELLOW
1.34	1.7	YELLOW
1.36	1.8	YELLOW
2.23	3.1	RED



x$activity　　x$size　　x$color

それぞれ、xの1列、2列、3列をベクトルとして抽出

よくある間違い

エラーが出た時にTAに聞く前に
まず、自分で確認してください！

- 大文字と小文字の書き間違い

  例. 正source(“…”)　誤Sourse(“…”)

- スペルミス

  例. 正source(“…”)　誤sourse(“…”)

- 0(ゼロ)とO(オー)、l(エル)とI(アイ)と1(数字)

  例. 正“1jl9.pdb”(1ジェイエル9)　誤”lj19.pdb”(エルジェイ19)

- 変更を保存せずに実行　→　変更が反映されません

  → **プログラムに変更を加える度に保存する(CTRL + S)癖をつけてください**

- ファイル名、ファイルの中では、なるべく**日本語/全角入力しない**こと

  → Rが正しくファイルを読み込んでくれません

  例. 正source(“…”)　誤source (“…”)

  ↑

  全角入力になっている

データフレームや行列では、行部分の条件式を書くとその条件を満たす行だけでできたデータフレームあるいは行列を作成できる

次のようにデータフレームdf を作成する

```
x <- c(1,2,2,3)
y <- c('A','A','C','B')
z <- c(100, 101, 102, 103)
df <- data.frame(X=x, Y=y, Z=z)
```

ここで

```
df[df$Y=='A',]
df[df$X==2, ]
```

と入力して、上の記述を確認しよう。また、この出力自体もデータフレームなので、

```
df[df$Y==‘A’,]$X
df[df$X==2, ]$Y
```

のようにして、得られた行の中の必要な要素だけを取り出すことができる。

4. 相同性

今回、相同なタンパク質の立体構造の
重ね合わせを行う

相同なタンパク質とは何か？

立体構造の重ね合わせには相同配列の
アラインメントが必要

相同配列の形成

・種分化　　ortholog

・遺伝子重複　　paralog

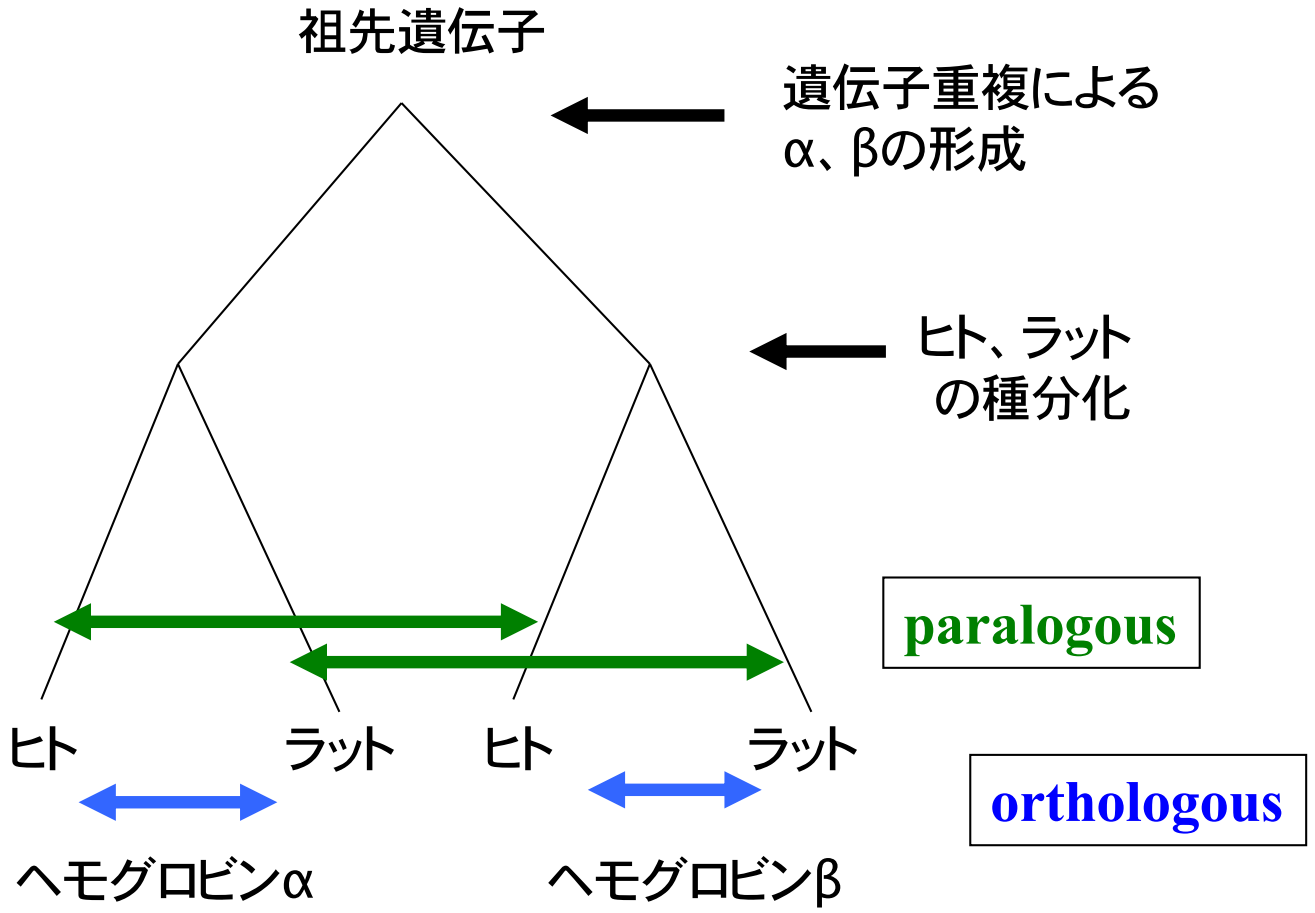
機能の多様化に特に重要

分子進化 (Molecular Evolution)

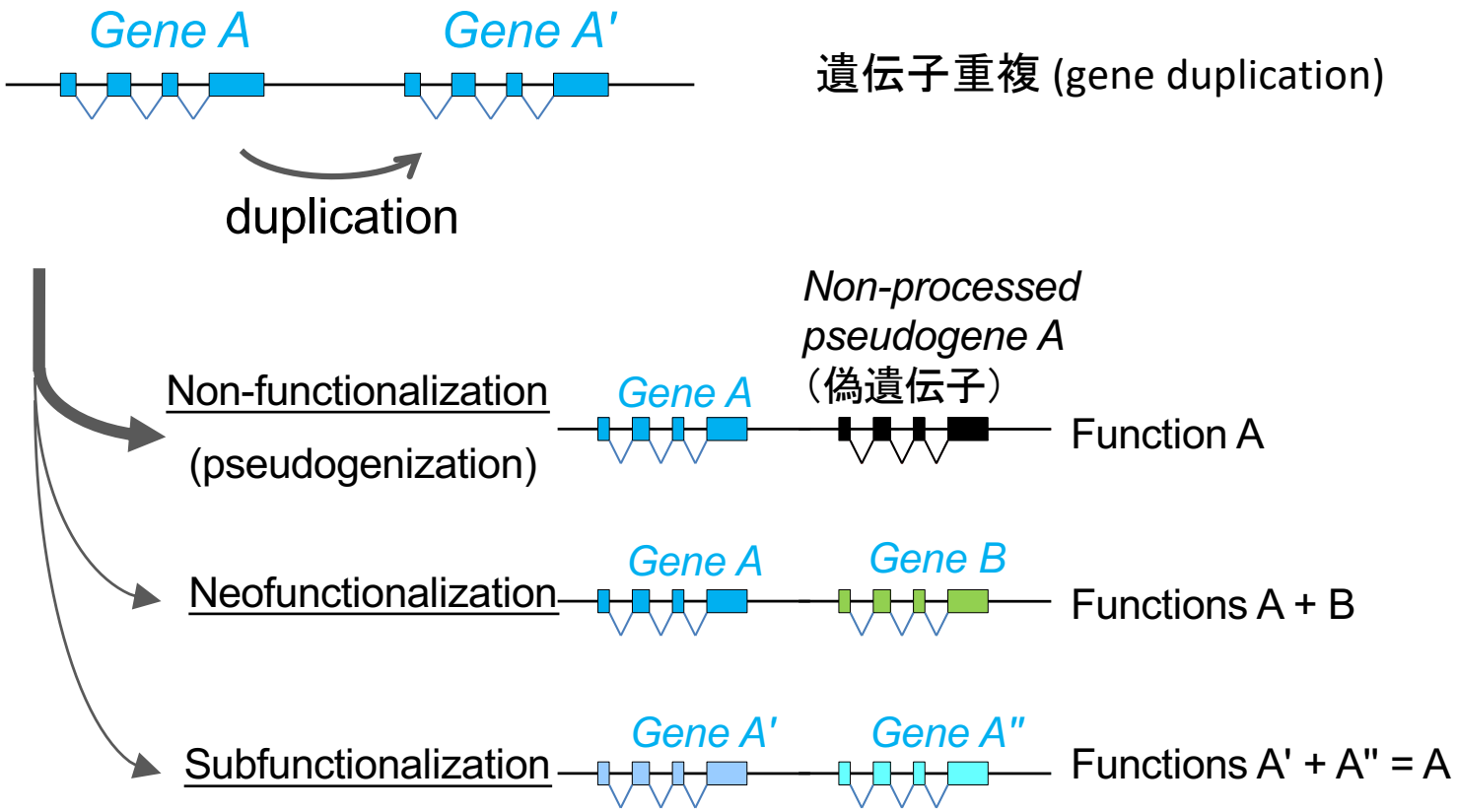
上の機構で分岐した遺伝子に突然変異が生じることで、配列が変化していくこと

基本ステップは

- ○ 塩基(アミノ酸)置換
- ○ 挿入/欠失

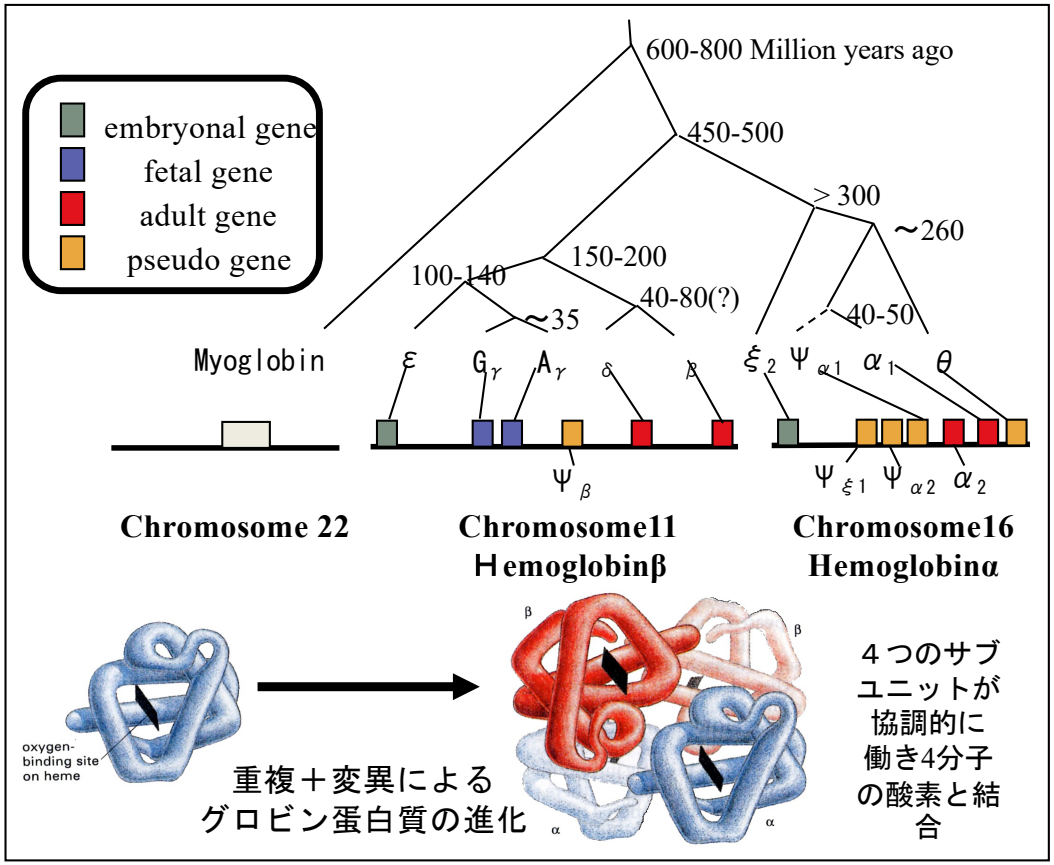

祖先遺伝子
遺伝子重複による
α、βの形成
ヒト、ラット
の種分化
paralogous
orthologous
ヒト
ラット
ヒト
ラット
ヘモグロビンα
ヘモグロビンβ

Evolutionary fate and functional consequence



●機能遺伝子の生成

・重複遺伝子が異なる機能を獲得した例（globin superfamily）



Fundamentals of Molecular evotution,
GENOMES second edition,
Molecular Biology of the Cell 4th editionを改変

＊Myoglobin：
・モノマーで機能

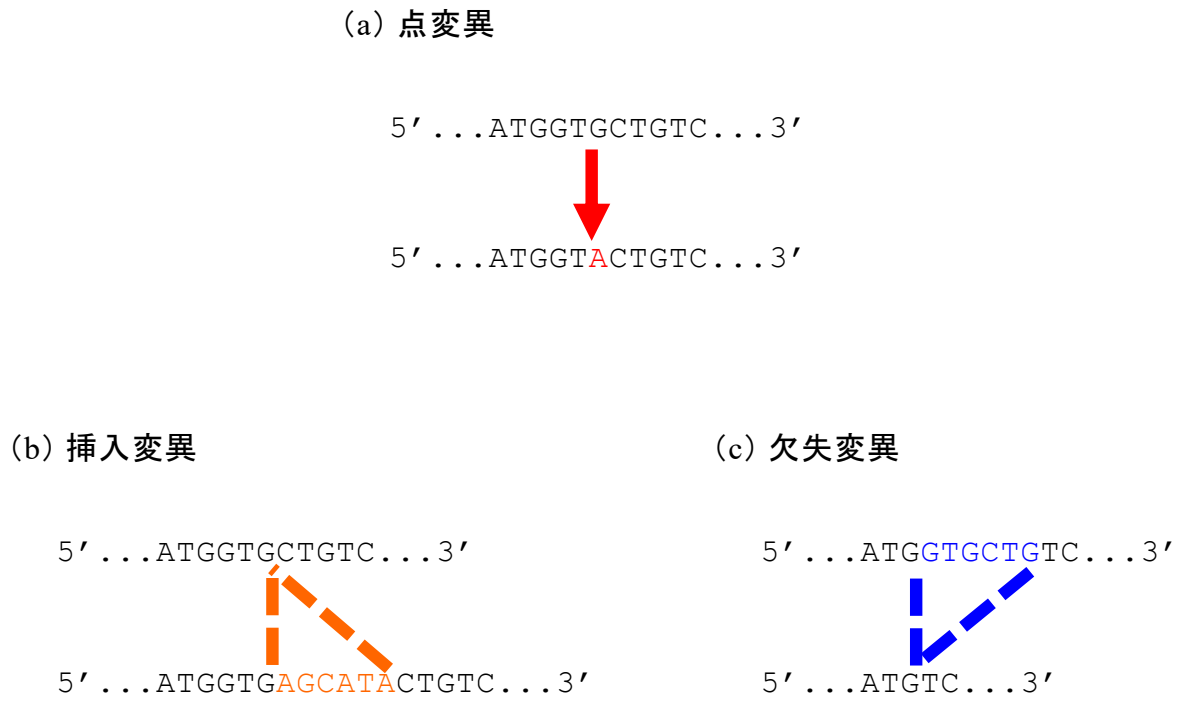
＊Hemoglobin：
・ヘテロテトラマーで機能
・各発生段階で発現するsubunitの組み合わせを変える

↓

酸素分子との親和性が変化し、段階に応じた機能が微調整される

ex.)
$\alpha_2\gamma_2$（胎児期）＞
$\alpha_2\beta_2$, $\alpha_2\delta_2$（成体）

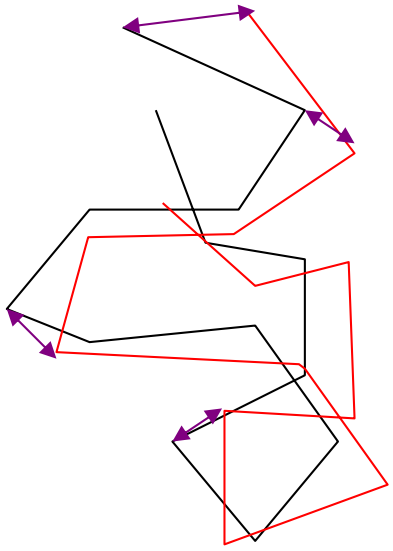
図1.5 点変異·挿入変異·欠失変異



RMSD

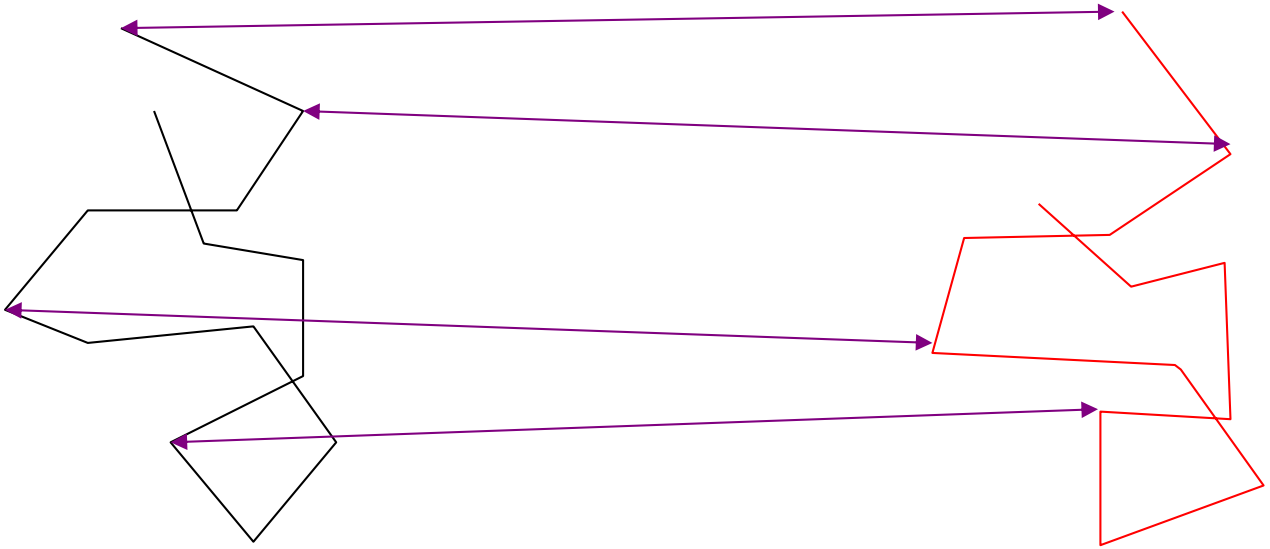
rmsd (root mean square distance) =

$$\sqrt{\frac{1}{n}\sum(dist(A(i),B(i))^2}$$



残基間対応が最初に
与えられていると
計算は容易

構造比較の原点　- 重ね合わせ (superposition) -



対応するCα原子間距離が最小になるように
二つの鎖を重ね合わせる (平行移動と回転)

McLachlan, A.D. (1972) *Nature New Biol*. 240, 83-85.

2. 配列アラインメント　(sequence alignment)

進化の過程でのINDELを考慮しながら、相同な配列の間の
対応する塩基（あるいはアミノ酸）を同じ位置に並べる操作
あるいは、その操作によってできたもの。

INDELに対応してギャップ(gap)とよばれる空記号を挿入し
位置をずらして、塩基やアミノ酸を対応づける。

通常、動的計画法(dynamic programing algorithm)や、その
バリエーションを用いて、配列間の類似度が高くとなるように
アラインメントが構築される。

アラインメントの原理は次回

マルチプルアラインメント　(multiple alignment)

分子系統樹を構築するには、まず複数本の相同配列
についてのマルチプルアラインメントを作成する。

リゾチームのアミノ酸配列

>LYC_HUMAN **ヒト** Lysozyme C
MKALIVLGLVLLSVTVQGKVFERCELARTLKRLGMDGYRGISLANWMCLAKWESGYNTRATNYNAGDRST
DYGIFQINSRYWCNDGKTPGAVNACHLSCSALLQDNIADAVACAKRVVRDPQGIRAWVAWRNRCQNRDVR
QYVQGCGV

>LYC1_BOVIN **ウシ** Lysozyme C 1
MKALIILGFLFLSVAVQGKVFERCELARTLKKLGLDGYKGVSLANWLCLTKWESSYNTKATNYNPGSEST
DYGIFQINSKWWCNDGKTPNAVDGCHVSCSELMENDIAKAVACAKQIVSEQGITAWVAWKSHCRDHDVSS
YVEGCTL

>LYC_CHICK **ニワトリ** Lysozyme C
MRSLLILVLCFLPLAALGKVFGRCELAAAMKRHGLDNYRGYSLGNWVCAAKFESNFNTQATNRNTDGSTD
YGILQINSRWWCNDGRTPGSRNLCNIPCSALLSSDITASVNCAKKIVSDGNGMNAWVAWRNRCKGTDVQA
WIRGCRL

>LYC2_ONCMY **マス** Lysozyme C II
MRAVVVLLLVAVASAKVYDRCELARALKASGMDGYAGNSLPNWVCLSKWESSYNTQATNRNTDGSTDYGI
FQINSRYWCDDGRTPGAKNVCGIRCSQLLTADLTVAIRCAKRVVLDPNGIGAWVAWRLHCQNQDLRSYVA
GCGV

>LYC_BOMMO **カイコ** Lysozyme
MQKLIIFALVVLCVGSEAKTFTRCGLVHELRKHGFEENLMRNWVCLVEHESSRDTSKTNTNRNGSKDYGL
FQINDRYWCSKGASPGKDCNVKCSDLLTDDITKAAKCAKKIYKRHRFDAWYGWKNHCQGSLPDISSC

>LYSP_DROME **ハエ** Lysozyme P
MKAFLVICALTLTAVATQARTMDRCSLAREMSKLGVPRDQLAKWTCIAQHESSFRTGVVGPANSNGSNDY
GIFQINNKYWCKPADGRFSYNECGLSCNALLTDDITNSVKCARKIQRQQGWTAWSTWKYCSGSLPSINSC
F

リゾチームのアミノ酸配列 マルチプル・アラインメント

二次構造情報を重ねて表示 α helix　β strand

ヒト
ウシ
マス
ニワトリ
カイコ
ハエ

アラインメントを作成してみよう

MAFFTによるマルチプルアラインメント

mafftは宮田研究室で開発され、加藤和貴によって継続的に
開発されているマルチプルアラインメントのフリーソフトウェア

海外の多くの研究機関で利用されている。

Web上でのアラインメントサービスに加え、ダウンロードして自身の
PC上で利用できる。Mac, Windows, Linuxなど様々なOSに対応している

ここでは、web serviceとして公開されているmafftを利用する。

https://mafft.cbrc.jp/alignment/software/

MAFFT version 7

Multiple alignment program for amino acid or nucleotide sequences

Download version

- Mac OS X
- Windows
- Linux
- Source

Online version

- Alignment
- mafft --add
- Merge
- Phylogeny
- Rough tree

Merits / limitations

Algorithms

Tips

Benchmarks

Feedback

Follow

Contact email address, kazutaka.katoh@aist.go.jp, is temporarily unavailable from 2018/Feb/7. If you sent an email to this address but have received no response, then please re-send the email to katoh@ifrec.osaka-u.ac.jp.

About

MAFFT is a multiple sequence alignment program for unix-like operating systems. It offers a range of multiple alignment methods, L-INS-i (accurate; for alignment of <~200 sequences), FFT-NS-2 (fast; for alignment of <~30,000 sequences), *etc.*

Download and Installation

- **Mac OS X**
- **Linux**
- **Windows**
- **Source**
- Changelog

The latest version is 7.471, 2020/Jul.

A bug in parsing input filename has been fixed. Please use 7.470 or higher. (2020/Jun)

Input Format

Fasta format. example1 (LSU rRNA), example2 (protein)

The type of input sequences (amino acid or nucleotide) is automatically recognized.

Usage

```
% mafft [arguments] input > output
```

An alias for an accurate option (L-INS-i) for an alignment of up to ~200 sequences × ~2,000 sites:

```
% mafft-linsi input > output
```

A fast option (FFT-NS-2) for a larger sequence alignment:

```
% mafft input > output
```

If not sure which option to use,

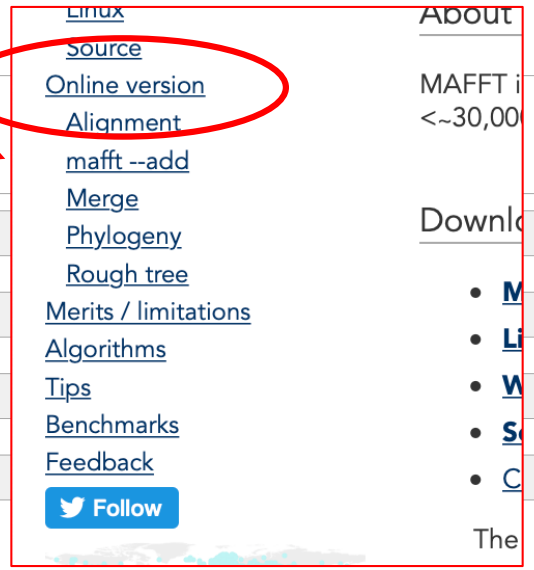
```
% mafft --auto input > output
```

- Manual (v6.240)
- Tips (not yet included in the manual) for large alignment, ncRNA alignment, profile alignment, *etc.*

Related Resources

- MAFFT server at EBI
- MAFFT server at the MPI Bioinformatics Toolkit
- ClustalW / MAFFT / PRRN at GenomeNet

Online Versionをクリック



MAFFT version 7

Multiple alignment program for amino acid or nucleotide sequences

CBRC AIST

Download version
- Mac OS X
- Windows
- Linux
- Source

Online version
- **Alignment**
- mafft --add
- Merge
- Phylogeny
- Rough tree

Merits / limitations
Algorithms
Tips
Benchmarks
Feedback

Follow

To avoid overload, try a light-weight option, for MSA of full-length SARS-CoV-2 genomes (2020/Apr).

For a large number of short sequences, try an experimental service.

Experimental service for aligning raw reads (2019/Aug)

Multiple sequence alignment and NJ / UPGMA phylogeny

Input:

Paste protein or DNA sequences in fasta format. Example

ファイルを選択をクリック

or upload a **plain text** file: ファイルを選択 ファイル未選択

Use DASH to add homologous structures (protein only) ***New! 2018/Dec/23***

Ouput original plus DASH sequences Output original sequences only

Give structural alignment(s) externally prepared

Allow unusual symbols (Selenocysteine "U", Inosine "i", non-alphabetical characters, *etc.*) Help

UPPERCASE / lowercase:

Same as input

Amino acid → UPPERCASE / Nucleotide → lowercase

Direction of nucleotide sequences: Help

Same as input

Adjust direction according to the first sequence (accurate enough for most cases)

Adjust direction according to the first sequence (only for highly divergent data; **extremely slow**)

Output order:

Same as input

Aligned

Job name (optional; used as output file name and subject of emails):

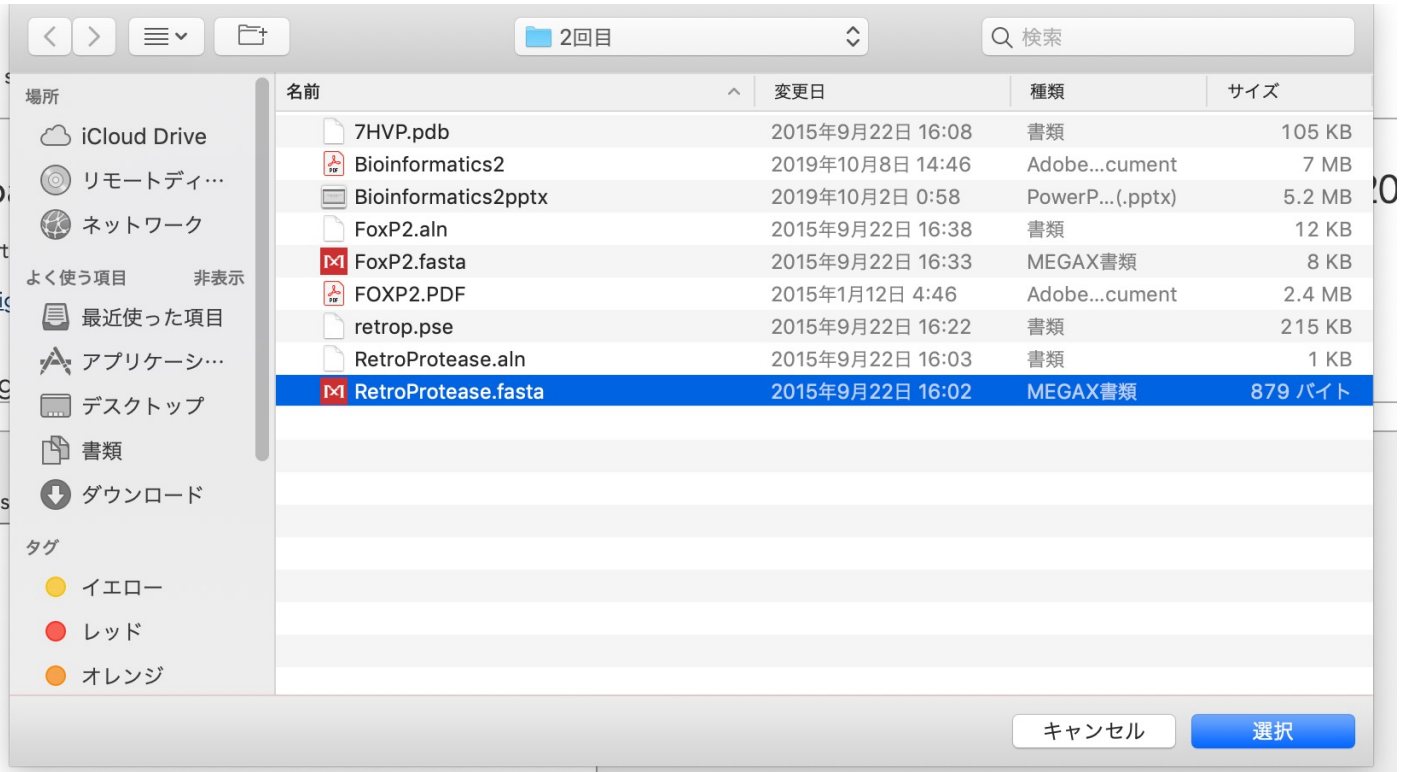
(basic Latin alphabet, number and space only)

Notify when finished (optional; recommended when submitting large data):

Email address:

Submit Reset

ファイルチューザからRetroProtease.fastaを選択



MAFFT version 7

Multiple alignment program for amino acid or nucleotide s

Download version
- Mac OS X
- Windows
- Linux
- Source

Online version
- **Alignment**
- mafft --add
- Merge
- Phylogeny
- Rough tree

Merits / limitations
Algorithms
Tips
Benchmarks
Feedback

Follow

To avoid overlo

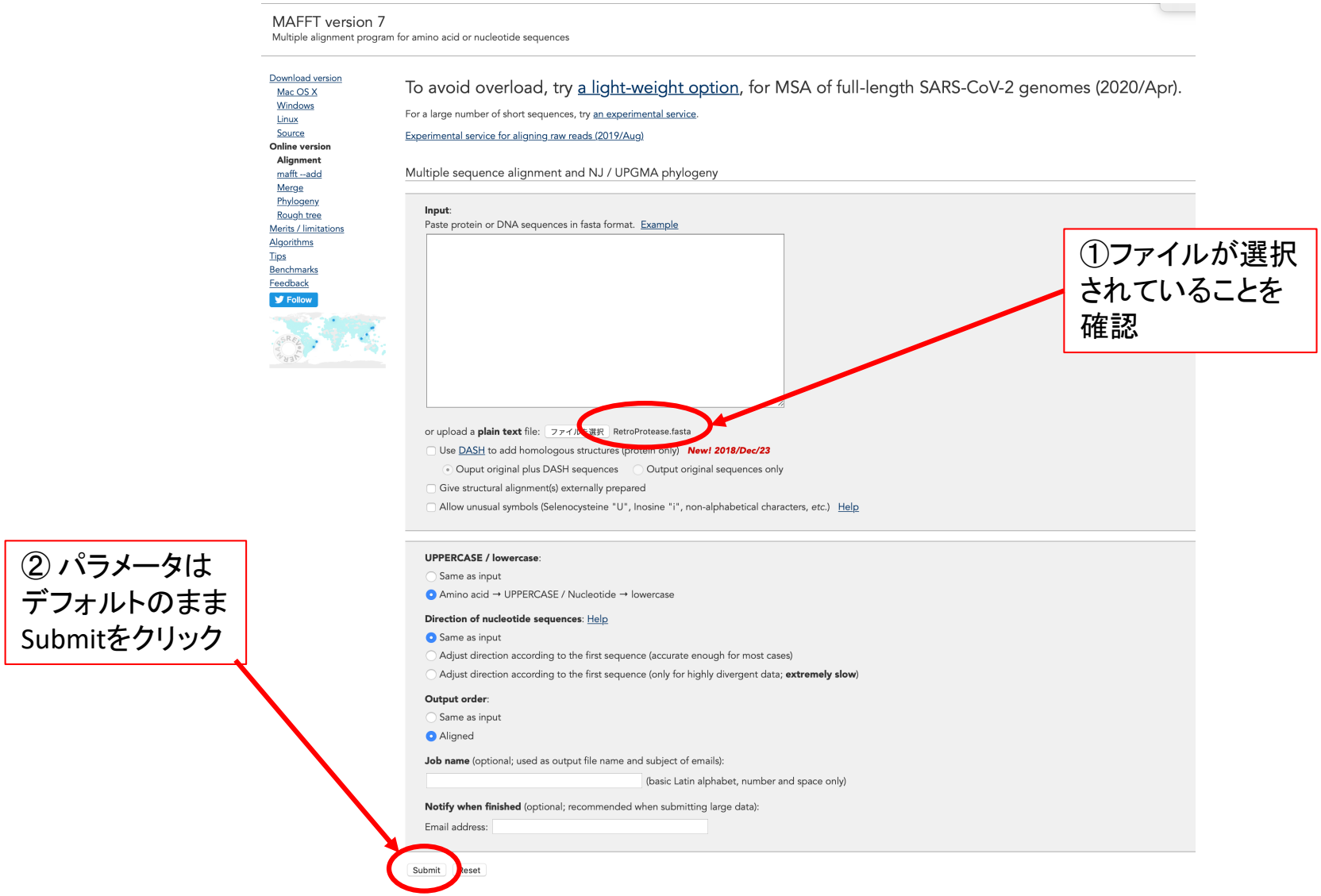
For a large number of short

Experimental service for alig

Multiple sequence alig

Input:
Paste protein or DNA s

or upload a **plain text** file: ファイルを選択 ファイル未選択


MAFFT version 7
Multiple alignment program for amino acid or nucleotide sequences
Download version
Mac OS X
Windows
Linux
Source
Online version
Alignment
mafft --add
Merge
Phylogeny
Rough tree
Merits / limitations
Algorithms
Tips
Benchmarks
Feedback
Follow
To avoid overload, try a light-weight option, for MSA of full-length SARS-CoV-2 genomes (2020/Apr).
For a large number of short sequences, try an experimental service.
Experimental service for aligning raw reads (2019/Aug)
Multiple sequence alignment and NJ / UPGMA phylogeny
Input:
Paste protein or DNA sequences in fasta format. Example
or upload a plain text file: ファイルを選択 RetroProtease.fasta
Use DASH to add homologous structures (protein only) New! 2018/Dec/23
Ouput original plus DASH sequences
Output original sequences only
Give structural alignment(s) externally prepared
Allow unusual symbols (Selenocysteine "U", Inosine "i", non-alphabetical characters, etc.) Help
UPPERCASE / lowercase:
Same as input
Amino acid → UPPERCASE / Nucleotide → lowercase
Direction of nucleotide sequences: Help
Same as input
Adjust direction according to the first sequence (accurate enough for most cases)
Adjust direction according to the first sequence (only for highly divergent data; extremely slow)
Output order:
Same as input
Aligned
Job name (optional; used as output file name and subject of emails):
(basic Latin alphabet, number and space only)
Notify when finished (optional; recommended when submitting large data):
Email address:
Submit
Reset
①ファイルが選択されていることを確認
② パラメータはデフォルトのままSubmitをクリック

CLUSTAL 形式
でアラインメント
が表示される

Clustal format | Fasta format | MAFFT result | View | Tree | Refine dataset | Return to home

View

Reformat to GCG, PHYLIP, MSF, NEXUS, uppercase/lowercase, *etc.* with Readseq

GUIDANCE2 computes the residue-wise confidence scores and extracts well-aligned residues.

Refine dataset

Phylogenetic tree

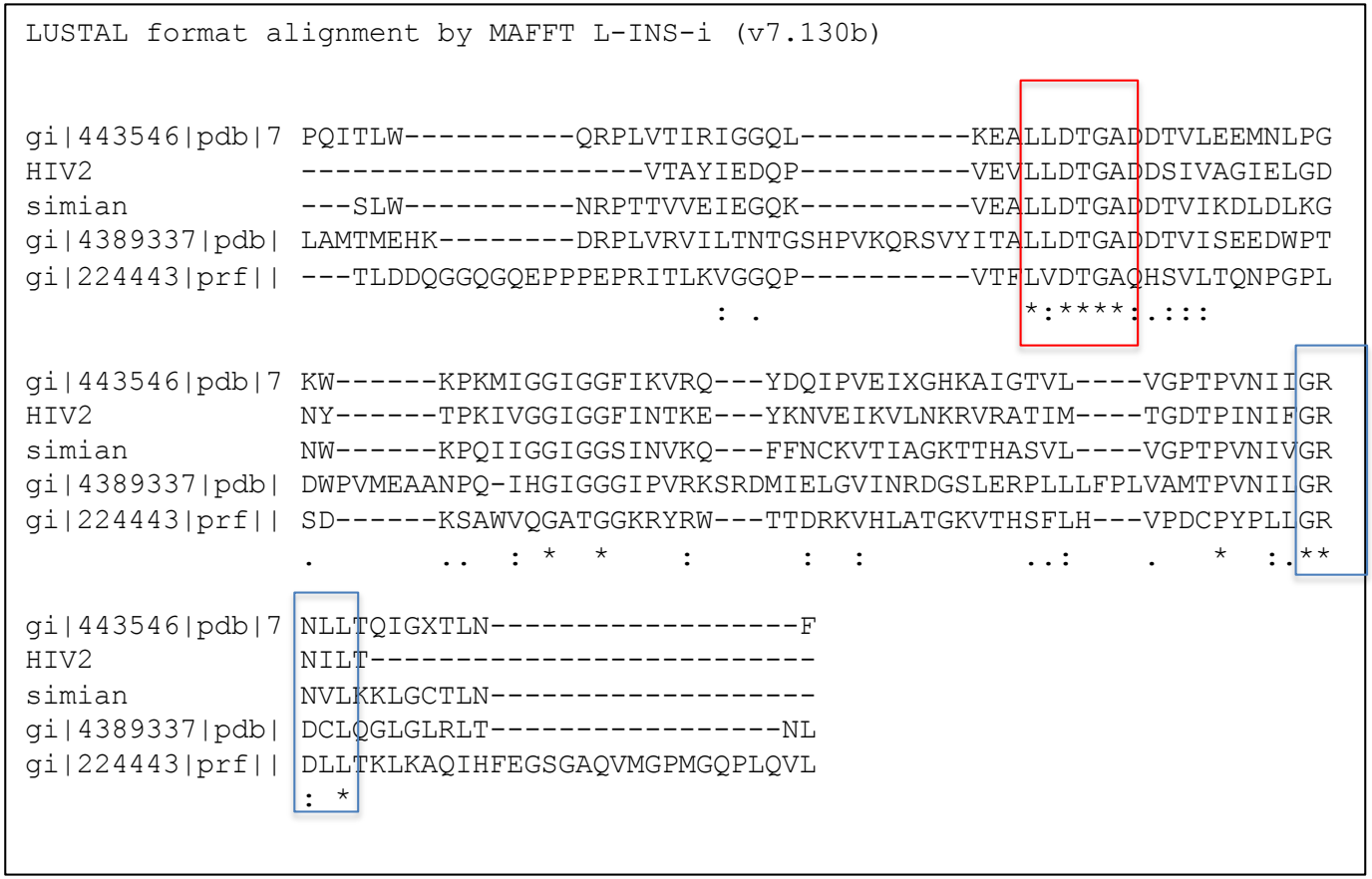
MAFFT-L-INS-i Result

```
CLUSTAL format alignment by MAFFT (v7.471)


gi|443546|pdb|7 PQITLW----------QRPLVTIRIGGQ----------LKEALLDTGADDTVLEEMNLPG
simian          ---SLW----------NRPTTVVEIEGQ----------KVEALLDTGADDTVIKDLDLKG
HIV2            --------------------VTAYIEDQ----------PVEVLLDTGADDSIVAGIELGD
gi|4389337|pdb| LAMTMEHK--------DRPLVRVILTNTGSHPVKQRSVYITALLDTGADDTVISEEDWPT
gi|224443|prf|| ---TLDDQGGQGQEPPPEPRITLKVGGQ----------PVTFLVDTGAQHSVLTQNPGPL
                                      : .              *:****:.:::

gi|443546|pdb|7 KW------KPKMIGGIGGFIKVRQ---YDQIPVEIXGHKAIGTVL----VGPTPVNIIGR
simian          NW------KPQIIGGIGGSINVKQ---FFNCKVTIAGKTTHASVL----VGPTPVNIVGR
HIV2            NY------TPKIVGGIGGFINTKE---YKNVEIKVLNKRVRATIM----TGDTPINIFGR
gi|4389337|pdb| DWPVMEAANPQ-IHGIGGGIPVRKSRDMIELGVINRDGSLERPLLLFPLVAMTPVNILGR
gi|224443|prf|| SD------KSAWVQGATGGKRYRW---TTDRKVHLATGKVTHSFLH---VPDCPYPLLGR
                .       ..  : *  *    :      :  :         ..:    .   *  :.**

gi|443546|pdb|7 NLLTQIGXTLN------------------F
simian          NVLKKLGCTLN------------------
HIV2            NILT-------------------------
gi|4389337|pdb| DCLQGLGLRLT----------------NL
gi|224443|prf|| DLLTKLKAQIHFEGSGAQVMGPMGQPLQVL
                : *
```

Method

L-INS-i (Probably most accurate, very slow)

```
% mafft --reorder --auto input
```



Clustal形式のアラインメント

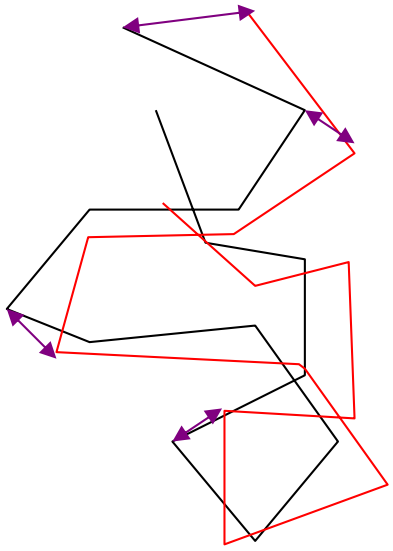
強く保存しているセグメント（モチーフ）が2ケ所見いだされる

5. 遺伝的アルゴリズムによる
相同タンパク質の立体構造の重ね合わせ

RMSD

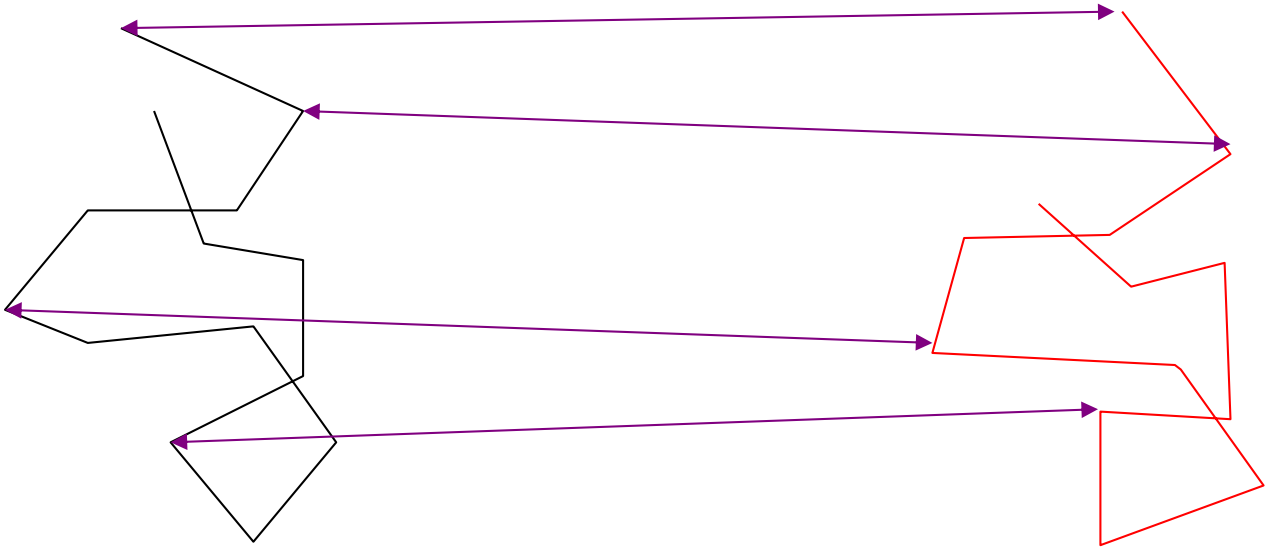
rmsd (root mean square distance) =

$$\sqrt{\frac{1}{n}\sum(dist(A(i),B(i))^2}$$



残基間対応が最初に
与えられていると
計算は容易

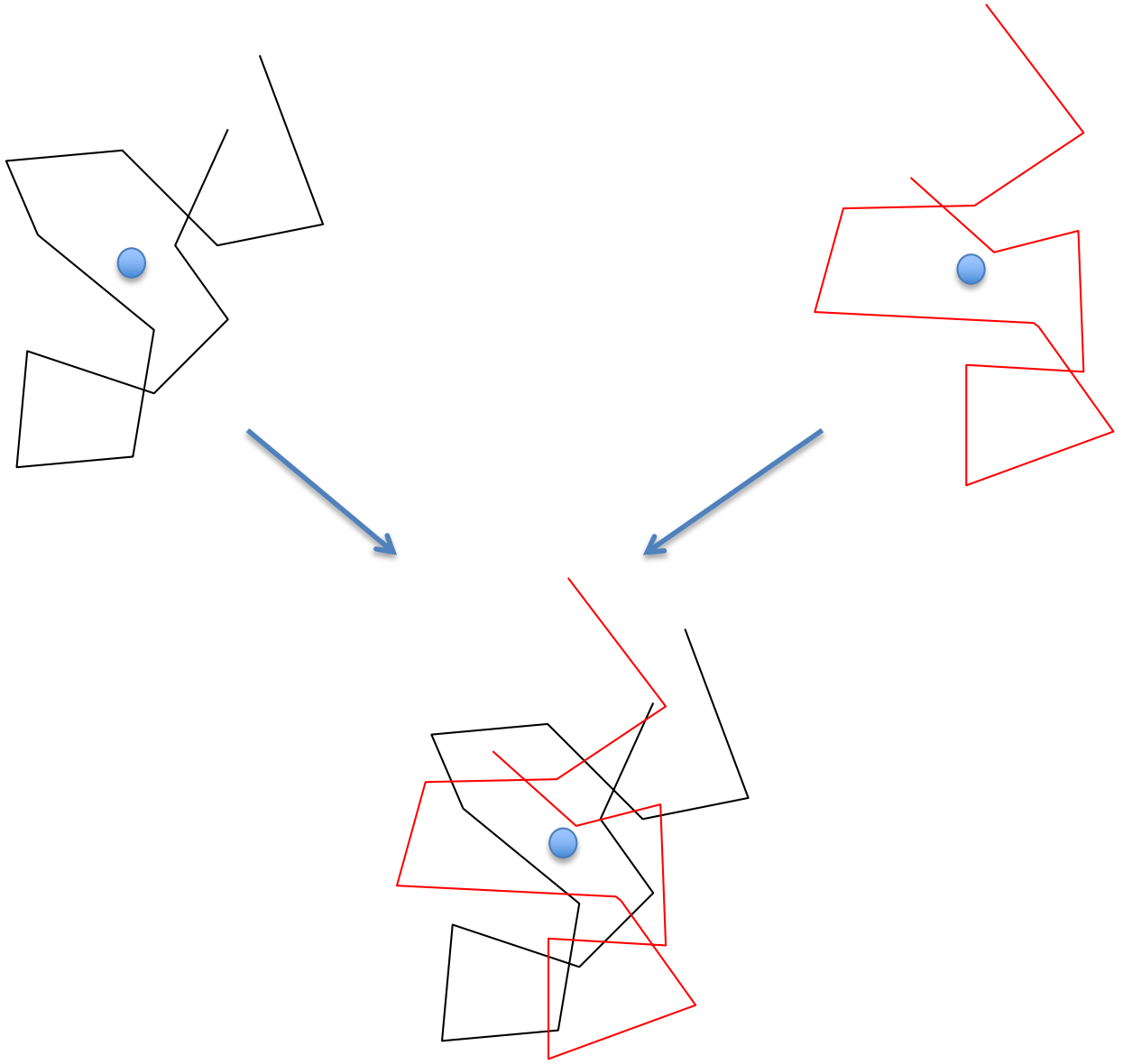
構造比較の原点　- 重ね合わせ (superposition) -

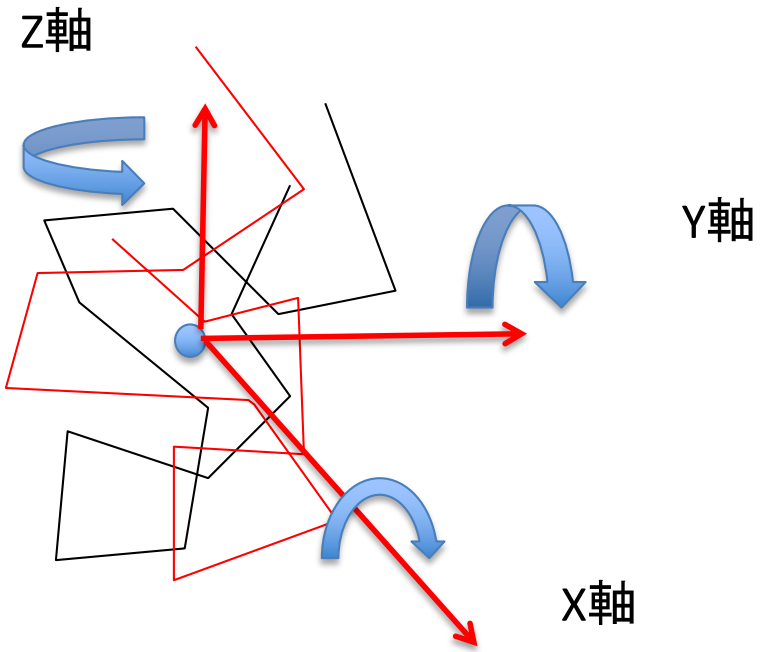


対応するCα原子間距離が最小になるように
二つの鎖を重ね合わせる (平行移動と回転)

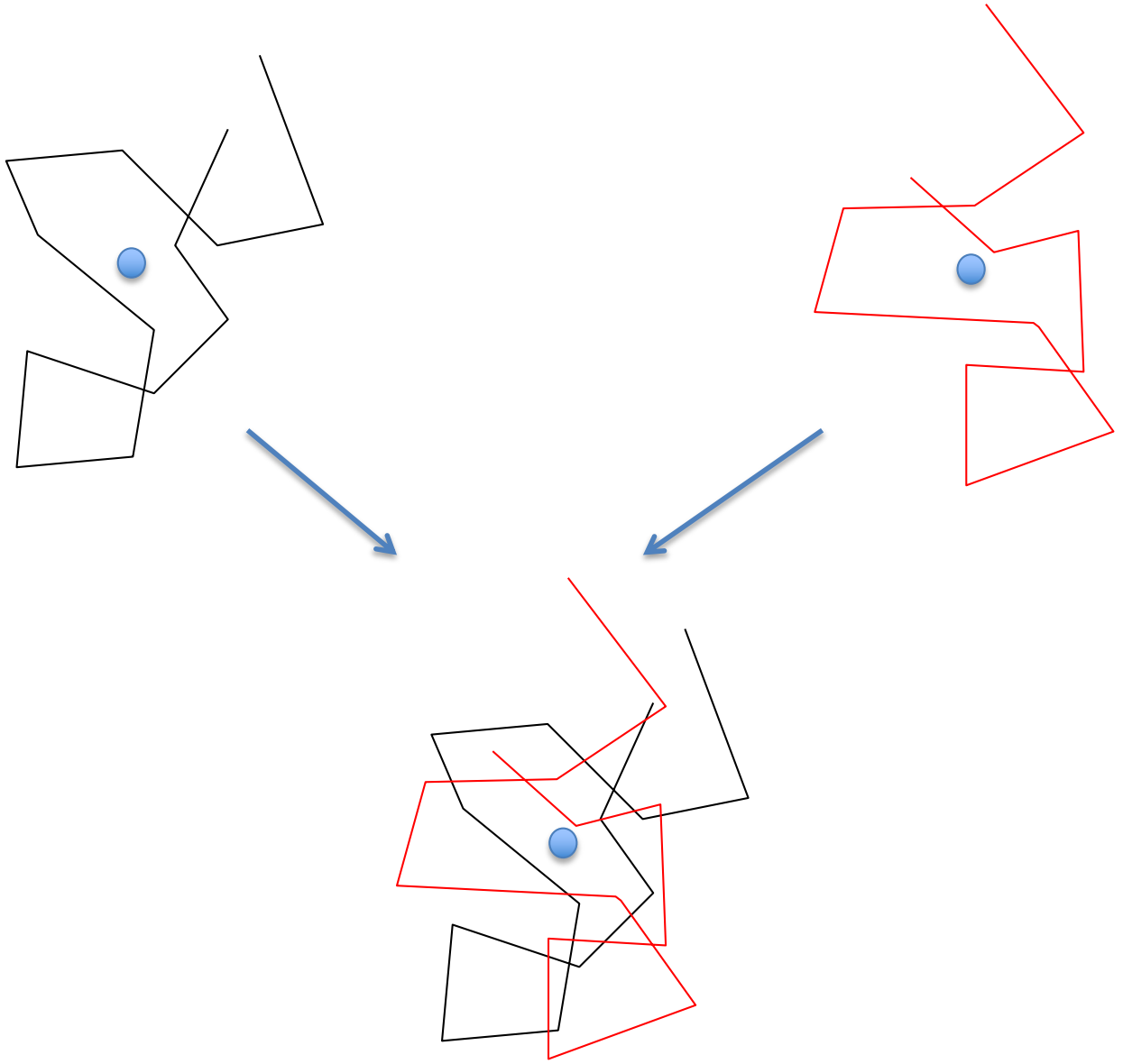
McLachlan, A.D. (1972) *Nature New Biol*. 240, 83-85.

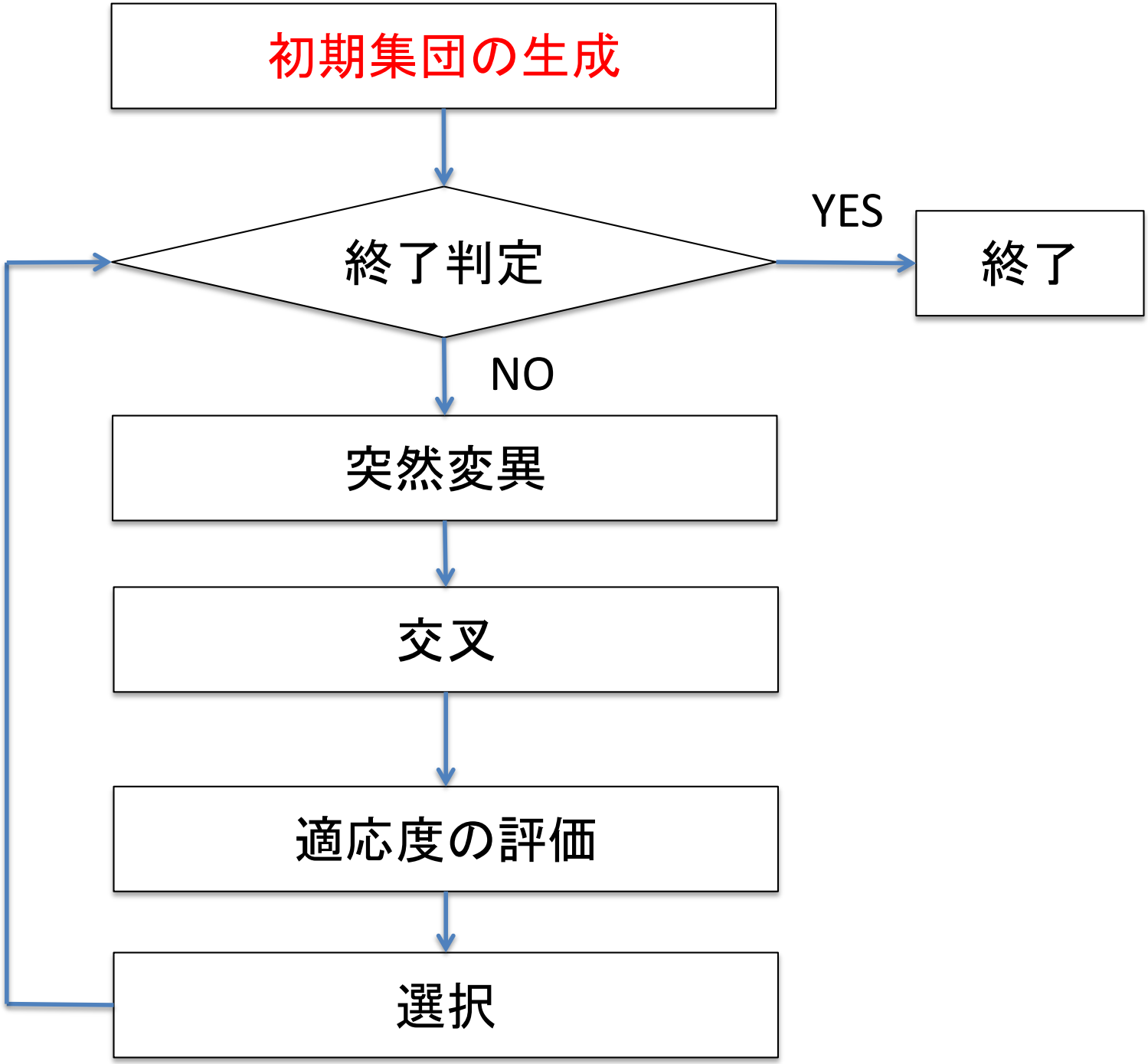
Cαの幾何重心を求め、それぞれ幾何重心の位置を原点とする




Z軸
Y軸
X軸

Cαの幾何重心を求め、それぞれ幾何重心の位置を原点とする




初期集団の生成
終了判定
YES
終了
NO
突然変異
交叉
適応度の評価
選択

遺伝子の設計

実数コード

遺伝子1

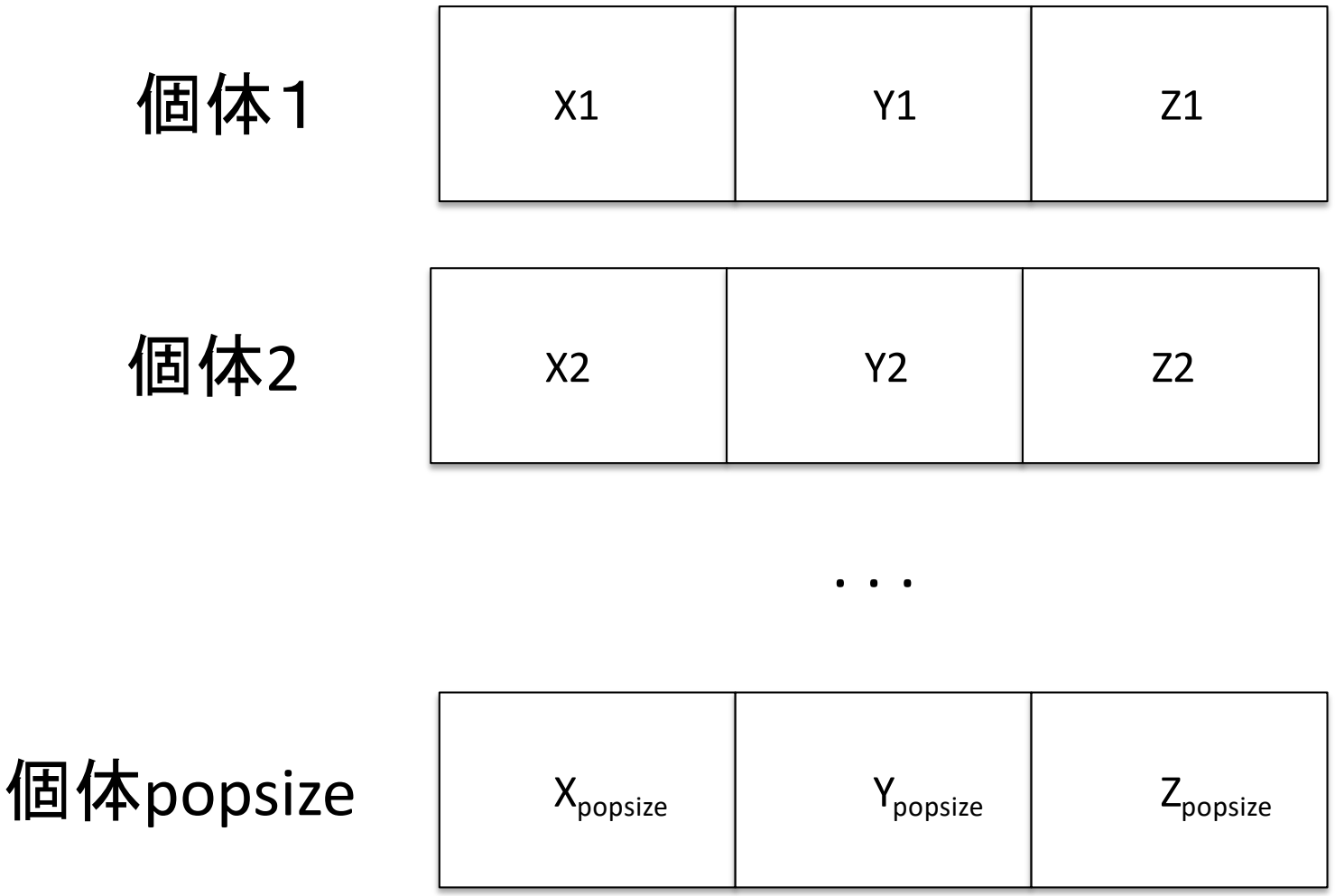
遺伝子2

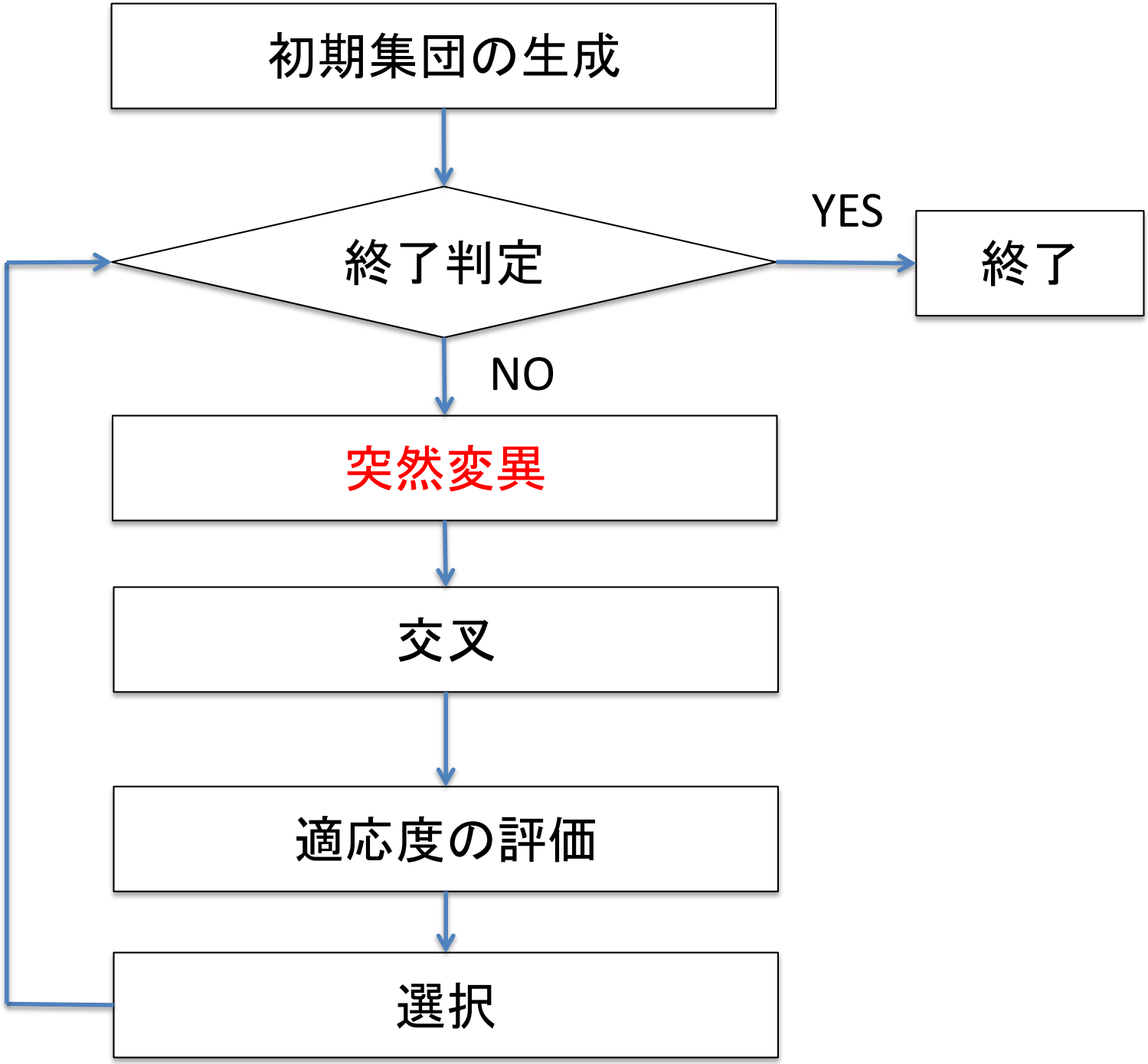
遺伝子3

染色体

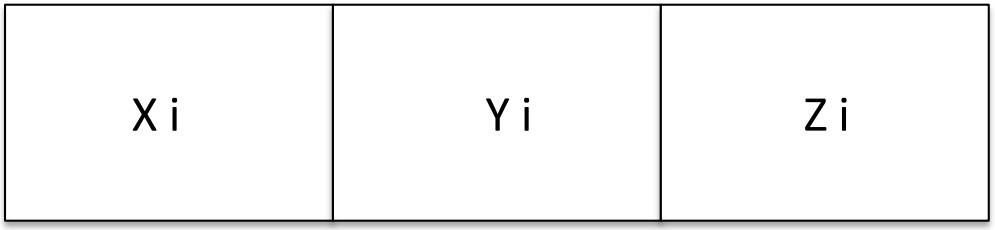
集団の生成

```
population <- matrix(runif(popsize*3)*2*pi,ncol=3)
```

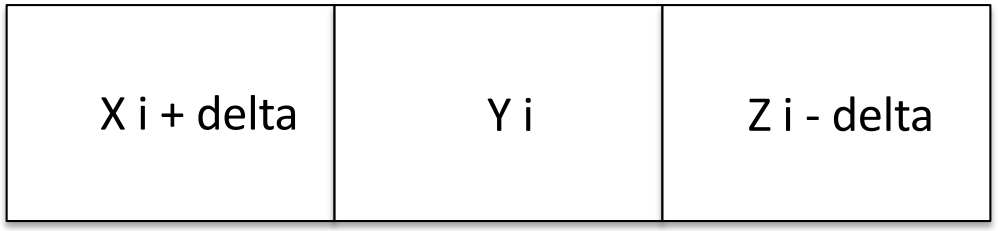


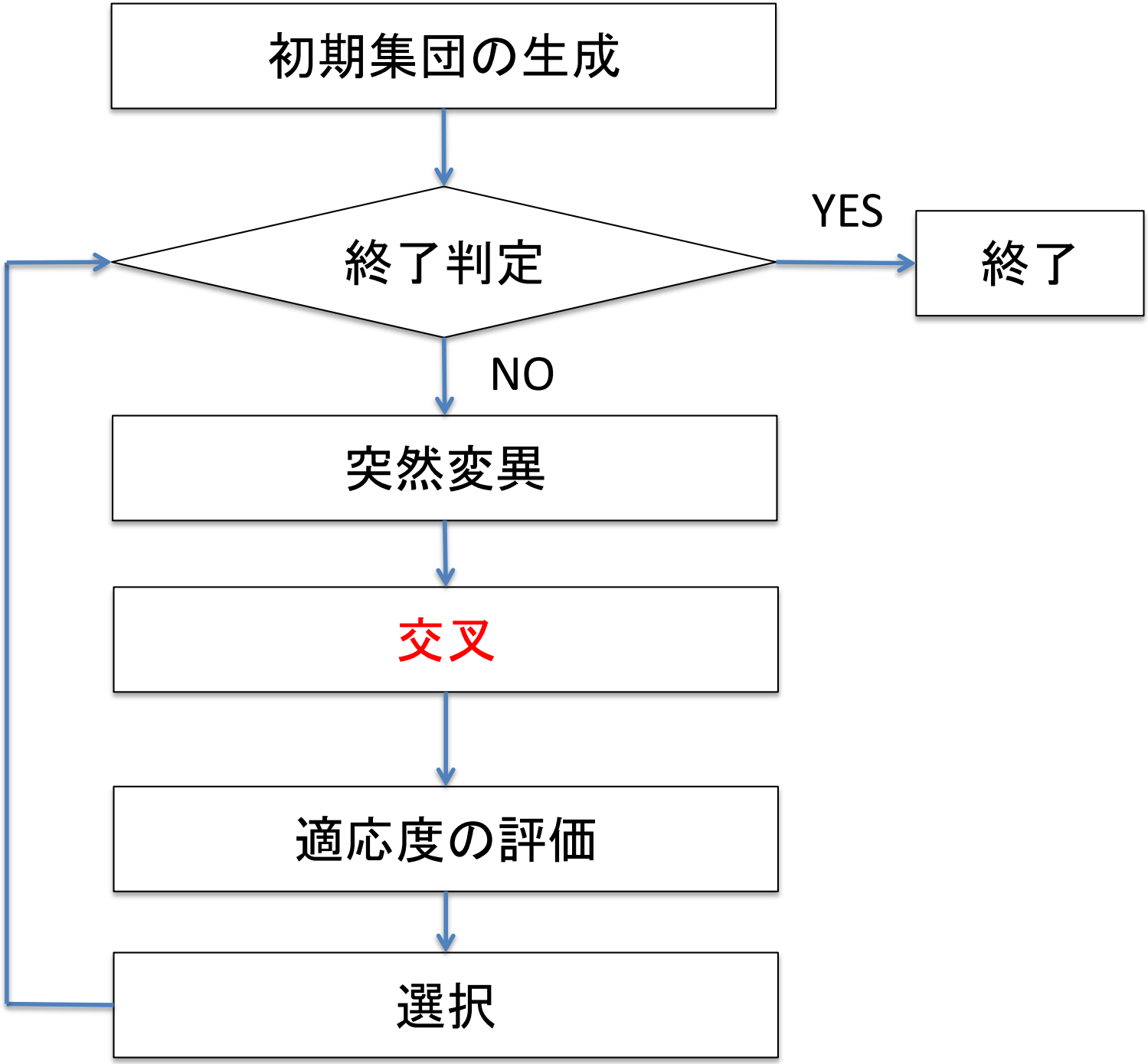

初期集団の生成
終了判定
YES
終了
NO
突然変異
交叉
適応度の評価
選択

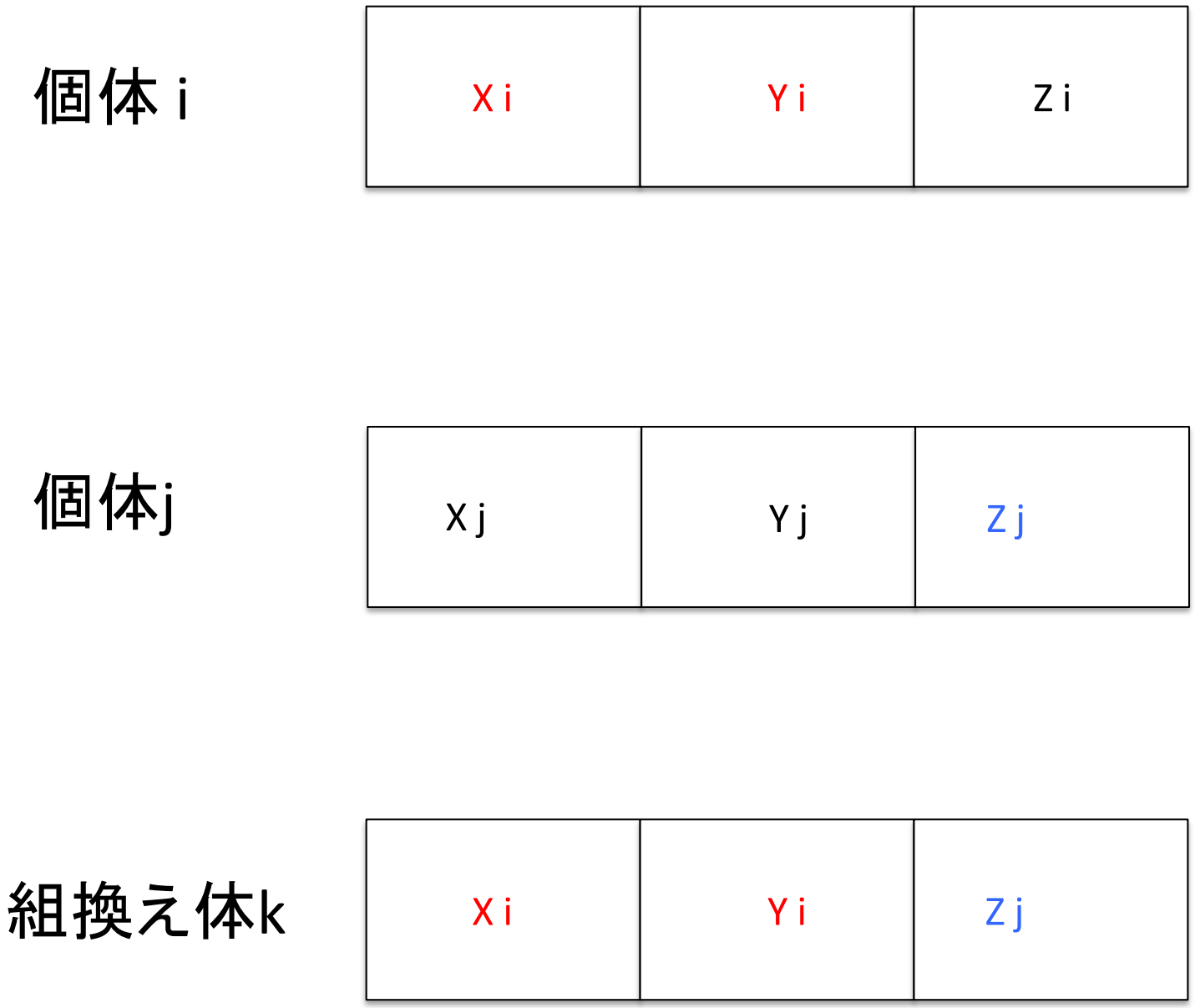
個体 i

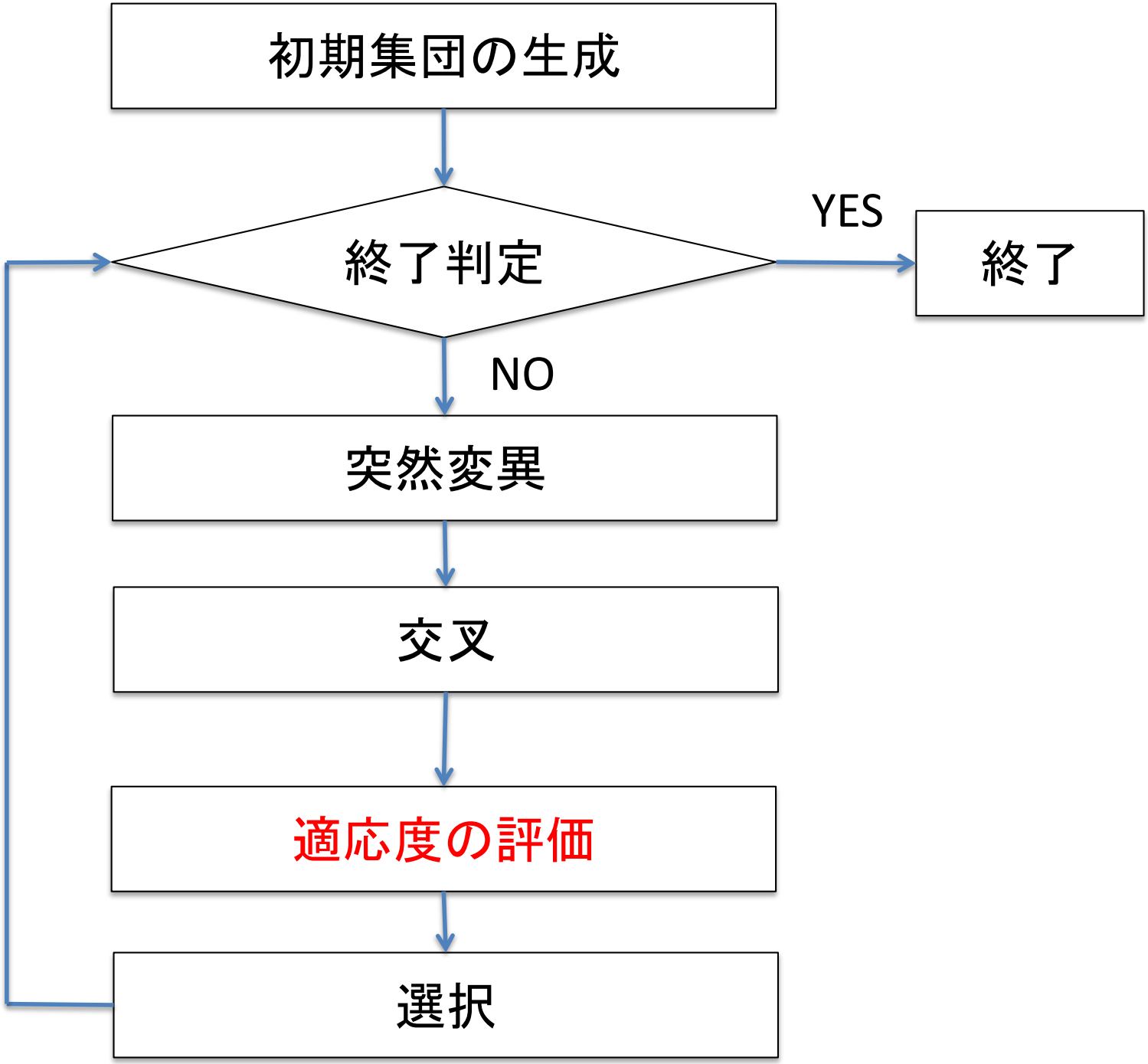


突然変異個体 j




初期集団の生成
終了判定
YES
終了
NO
突然変異
交叉
適応度の評価
選択


個体 i
X i
Y i
Z i
個体j
X j
Y j
Z j
組換え体k
X i
Y i
Z j


初期集団の生成
終了判定
YES
終了
NO
突然変異
交叉
適応度の評価
選択

population

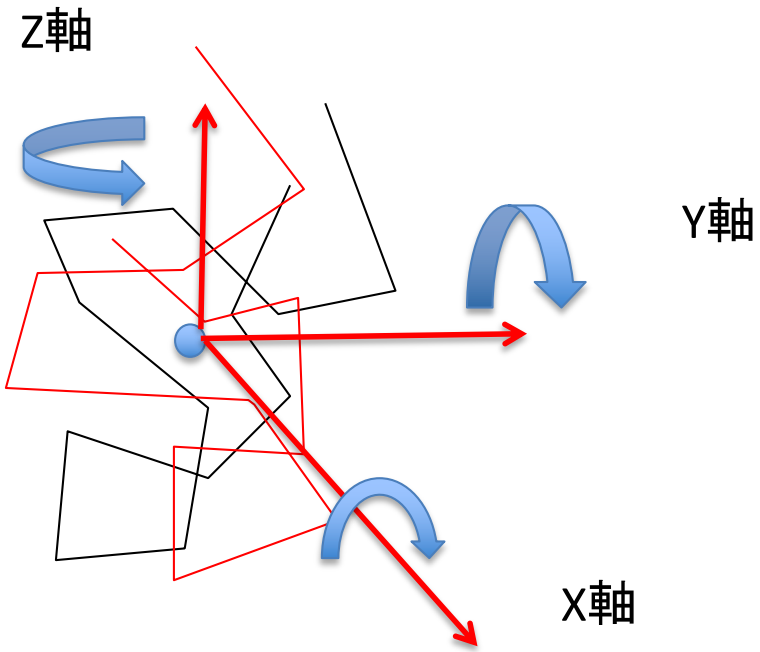
mutants

recombinants

X1	Y1	Z1
X2	Y2	Z2
Xpopsize	Ypopsize	Zpopsize
X'1	Y'1	Z'1
X'msize	Y'msize	Z'msize
X''1	Y''1	Z''1
X''rsize	Y''size	Z''rsize

それぞれに対してfitness
を計算

$$\text{Fitness} = \frac{1}{\text{各個体の回転操作による構造間のRMSD} + 0.1}$$


Z軸
Y軸
X軸

3次元の回転行列 [編集]

各軸周りの回転 [編集]

3次元空間での*x*軸、*y*軸、*z*軸周りの回転を表す回転行列は、それぞれ次の通りである：

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix}$$

$$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix}$$

$$R_z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

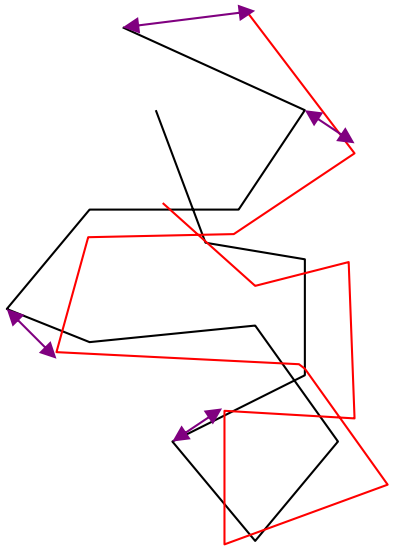
ここで回転の方向は、R_x は*y*軸を*z*軸に向ける方向、R_y は*z*軸を*x*軸に向ける方向、R_z は*x*軸を*y*軸に向ける方向である。

`rx, ry, rz` は、それぞれx軸、y軸、z軸の周りで、引数として与えられた角度だけ回転させる回転行列を作成し、それを返り値とする関数
`FunctionLibrary.R` 中に記述されており、`output`と`calcFitness`の二つの関数で使用されている。

RMSD

rmsd (root mean square distance) =

$$\sqrt{\frac{1}{n}\sum(dist(A(i),B(i))^2}$$



残基間対応が最初に
与えられていると
計算は容易

注意事項

今回はGAを使用した発見的な方法で重ねあわせを行っているが、重ねあわせは
厳密に求める方法があるので、実際の使用にあたっては、そちらを利用する方が良い

今回はGAを理解する手段として重ねあわせを用いてるだけである。

課題の説明

[1] 1jl9.pdbと1jl9m.pdbの重ね合わせ
[1-1] GAに及ぼす集団サイズの効果
[1-2] GAに及ぼす世代数の効果
[1-3] GAに及ぼす突然変異率の効果
[1-4] GAに及ぼす組換え率の効果

[2] 1jl9.pdbと3c9a.pdbの重ね合わせ
[2-1] GAに及ぼす集団サイズの効果
[2-2] GAに及ぼす世代数の効果
[2-3] GAに及ぼす突然変異率の効果
[2-4] GAに及ぼす組換え率の効果

[3] プログラムの修正による1jl9.pdbと3c9a.pdbの重ね合わせの改良
[3-1] 修正前に実行
[3-2] 修正後に実行

重ね合わせデータ

今回、

(1)1jl9.pdbと1jl9m.pdbの重ね合わせ

と

(2) 1jl9.pdbと3c9a.pdbの重ね合わせ

を行う。

1jl9.pdb　human epidermal growth factor (EGF)
　42アミノ酸

3c9a.pdb　*Drosophila melanogaster* Spitzタンパク質の
　EGFドメイン
　48アミノ酸

重ね合わせには、一方の構造のどのアミノ酸と他方の構造のどのアミノ酸が対応するかを決めておかないと、RMSDが計算できない。

対応関係はアラインメントファイルで与える。

アラインメント

(1)　1jl9.pdbと1jl9m.pdbの重ね合わせ用

1jl9.aln.fasta

```
>1JL9A:EPIDERMAL GROWTH FACTOR
CPLSHDGYCLHDGVCMYIEALDKYACNCVVGYIGERCQYRDL
>1JL9B:EPIDERMAL GROWTH FACTOR
CPLSHDGYCLHDGVCMYIEALDKYACNCVVGYIGERCQYRDL
```

(2) 1jl9.pdbと3c9a.pdbの重ね合わせ用

seq.aln.fasta

```
>3C9A:C|PDBID|CHAIN|SEQUENCE
PTYKCPETFDAWYCLNDAHCFAVKIADLPVYSCECAIGFMGQRCEYKE-
>1JL9B:EPIDERMAL GROWTH FACTOR
----CPLSHDG-YCLHDGVCMYIEALD--KYACNCVVGYIGERCQYRDL
```

進化の過程での挿入/欠失を考慮して、配列間の類似度が最大になるように並置
上記のファイル形式をFASTA形式と呼ぶ(“>”で始まる行は注釈行、それ以外は配列データ)
‘-’はギャップと呼ばれる空記号。挿入/欠失に対応させて、アミノ酸の位置をずらすために使われる。
赤:一致、青:置換されているが物理化学的に類似

使用するデータファイル

（1）立体構造

1jl9.pdb, 1jl9m.pdb, 3c9a.pdb

の3つ

（2）アラインメント

1jl9.aln.fasta、seq.aln.fasta

の2つ

テキスト形式で、改行コードはWindowsにしてある

もし、MacやLinuxユーザでファイルの改行がおかしかったり、Rでうまく読み込めない場合は、藤(tohhir@kwansei.ac.jp)まで連絡すること

プログラム

ExecuteGA.R

FunctionLibraryX.R

FunctionLibraryY.R

ExecuteGA.R

第一パート

GAのパラメータや入力ファイルや出力ファイルを設定する。

第二パート

GAの実施部分

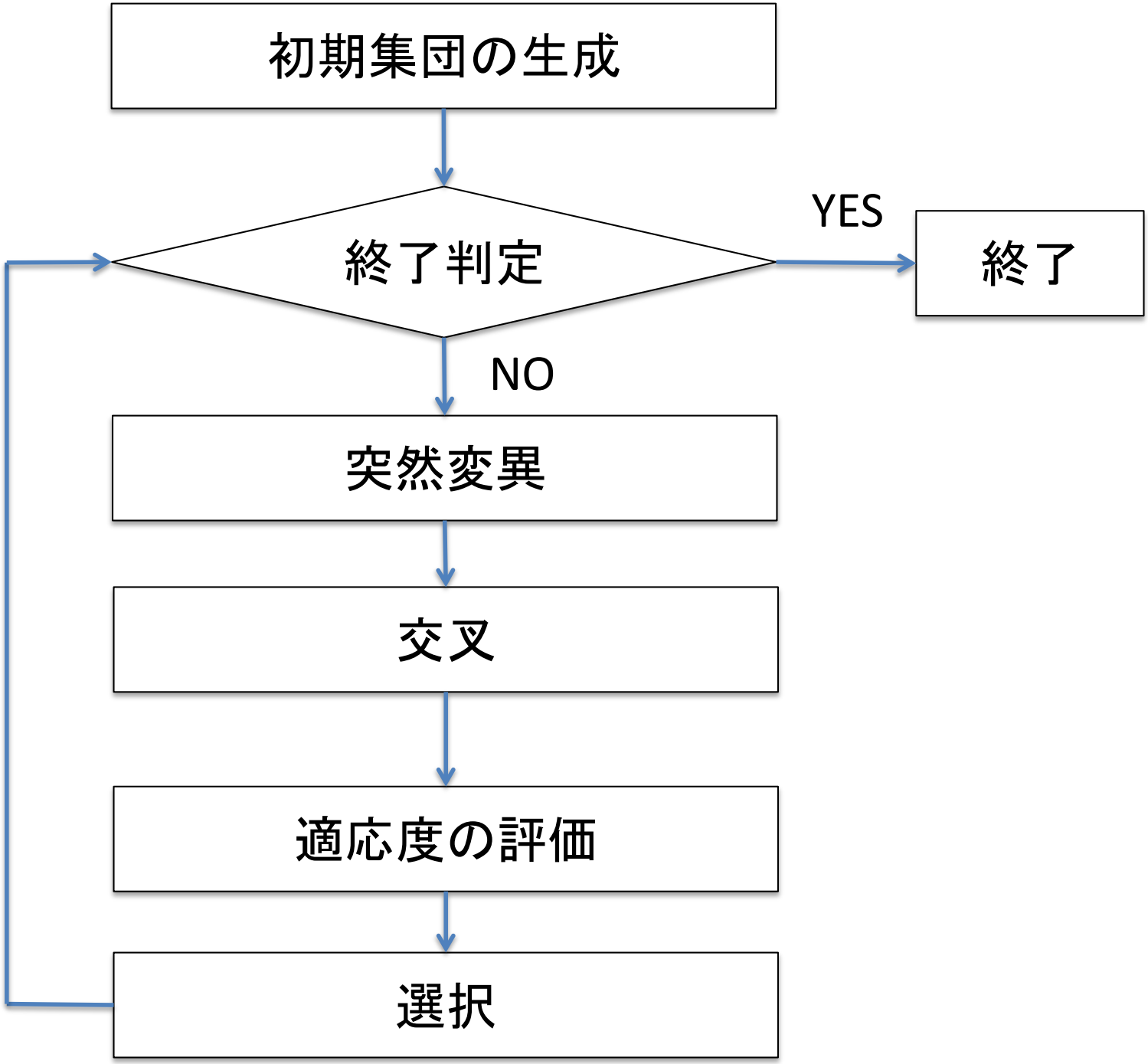
第一パートの構造データの指定の順番と
アラインメントデータの順番が一致するように気をつけること

seq.aln.fastaでは、アラインメントが

```
>3C9A:C|PDBID|CHAIN|SEQUENCE
PTYKCPETFDAWYCLNDAHCFAVKIADLPVYSCECAIGFMGQRCEYKE-
>1JL9B:EPIDERMAL GROWTH FACTOR
----CPLSHDG-YCLHDGVCMYIEALD—KYACNCVVGYIGERCQYRDL
```

と与えられている。この時、ExecuteGA.Rでは、

```
#入力立体構造1
InputFileName1 <- '3c9a.pdb'
#入力立体構造2
InputFileName2 <- '1jl9m.pdb'
#アラインメントファイル
AlignmentFile <- 'seq.aln.fasta'
#変異率
MutationRate <- 0.9
#組み換え率
RecombinationRate <- 0.9
#redex:適応度の更新が見られなかった時に、どれくらい回転角度の範囲を狭めるのかを示すパラメータ
redx <- 1.0
#出力ファイル名1(回転後の立体構造1)
OutputFileName1 <- '3c9a_500_100_0.9_0.9_1.pdb'
#出力ファイル名2(回転後の立体構造2)
OutputFileName2 <- '1jl9_500_100_0.9_0.9_1.pdb'
#出力ファイル名3(経過プロット)
OutputPlotName <- 'plot_3c9a_1jl9_500_100_0.9_0.9_1.png'
```


初期集団の生成
終了判定
YES
終了
NO
突然変異
交叉
適応度の評価
選択

第一パート

```
#集団サイズ　　このパラメータを書き換えます
PopulationSize <- 10

#世代数　　このパラメータを書き換えます
GenerationNumber <- 100

#入力立体構造1　　この入力ファイル名を書き換えます
InputFileName1 <- '~/Desktop/先端実習2020/1jl9.pdb'

#入力立体構造2　　この入力ファイル名を書き換えます
InputFileName2 <- '~/Desktop/先端実習2020/1jl9m.pdb'

#アラインメントファイル　　この入力ファイル名を書き換えます
AlignmentFile <- '~/Desktop/先端実習2020/1jl9.aln.fasta'

#変異率　　このパラメータを書き換えます
MutationRate <- 1.0

#組み換え率　　このパラメータを書き換えます。
RecombinationRate <- 1.0

#redex:適応度の更新が見られなかった時に、どれくらい回転角度の範囲を狭めるのかを示すパラメー
redx <- 1.0　　#　固定にします、今回は使いません。
```

第一パート (続き)

#出力ファイル名1 (回転後の立体構造1) この出力ファイル名を書き換える(設定したパラメータを反映する名前にする
```
OutputFileName1 <- '1jl9_10_100_1.0_1.0.pdb'
```

#出力ファイル名2 (回転後の立体構造2) この出力ファイル名を書き換える(設定したパラメータを反映する名前にする
```
OutputFileName2 <- '1jl9m_10_100_1.0_1.0.pdb'
```

#出力ファイル名3 (経過プロット) この出力ファイル名を書き換える (設定したパラメータを反映する名前にする
```
OutputPlotName <- 'plot_1jl9_1jl9m_10_100_1.0_1.0.png'
```

注意:同じパラメータで実行する際には、出力ファイル名を変更すること。
変更しないと読み込み時に停止する (関数prepareGAの中の処理)

ExecuteGA.Rを実行する際の注意

- 今回の実習の後半で扱う**遺伝的アルゴリズムのプログラム**は、出力ファイルが**上書きされないように**、すでにフォルダにあるファイルと同じファイル名を出力しようとすると、下のような**エラーが出る**ようにしています

コンソール画面：

```
> source('~/Desktop/2017 sentan/Rprograms/ExecuteGA.R', chdir = TRUE)
Error in prepareGA(InputFileName1, InputFileName2, AlignmentFile, PopulationSize) :
  ERROR::Same OutputFileName  modified1jl9.pdb  was detected in this directory...
 This Program have been stopped.  Rename Your OutputfileName1.
```

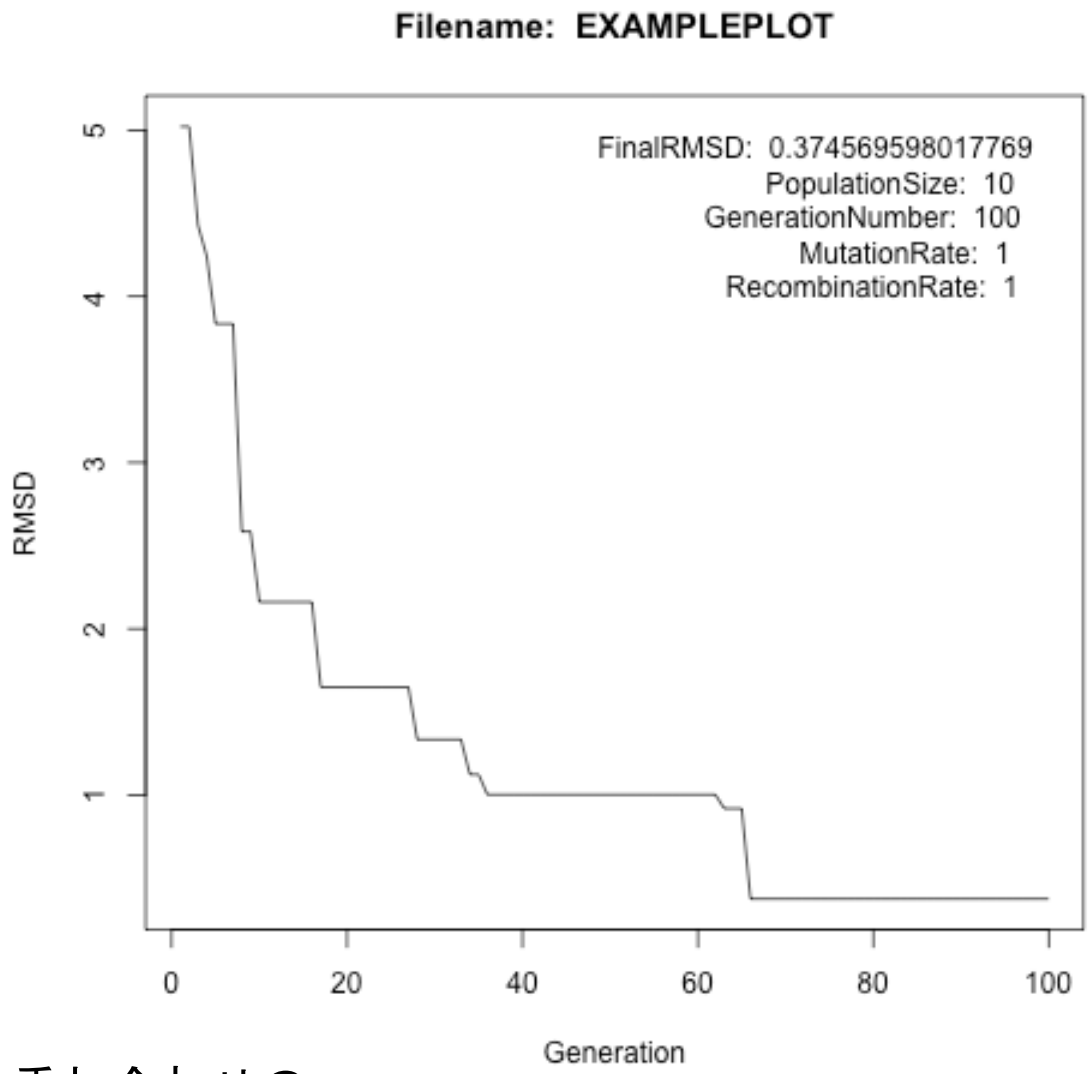
→ ExecuteGA.Rを実行する度に、ExecuteGA.Rのプログラムの**出力するファイル名を変更してください**

第二パート

```
# 関数ライブラリを読み込みます。
source("FunctionLibraryY.R")

# 遺伝的アルゴリズム構築##########################################
# 以下の preparationGA, mutation, recombination, calculationFitness, selection,
# output はFunctionLibraryY.Rの中で定義されている関数
# 初期集団を作成します。
Population <- prepareGA(InputFileName1, InputFileName2, AlignmentFile,
                        PopulationSize)

# 繰り返し操作を実行します。
for(i in 1:GenerationNumber){
  # 突然変異を行います。
  Mutated <- mutation(Population, MutationRate)
  ## Mutatedには、親集団と突然変異で作成した子集団を一緒にした集団が格納
  # 組み換え(交叉)を行います。
  Recombinated <- recombination(Mutated, RecombinationRate)
  ## Recombinatedには、MutatedとMutatedから組換えで作成した子集団を一枝にした集団が格納
  # 適応度の評価を行います。
  Fitness <- calculationFitness(Recombinated)
  ## 集団recombinatedの各個体に対する適応度(fitness)を計算
  # 適応度にしたがって個体の選択を行います。
  Population <- selection(Recombinated, Fitness)
  ## Fitnessに従いRecombinatedから次世代のPopulationを作成する
}

# 終了結果をファイルに出力します。
output(Population, Fitness, OutputFileName1, OutputFileName2, OutputPlotName)
```

出力例

出力ファイル名3で指定されたpngファイル



横軸:世代数

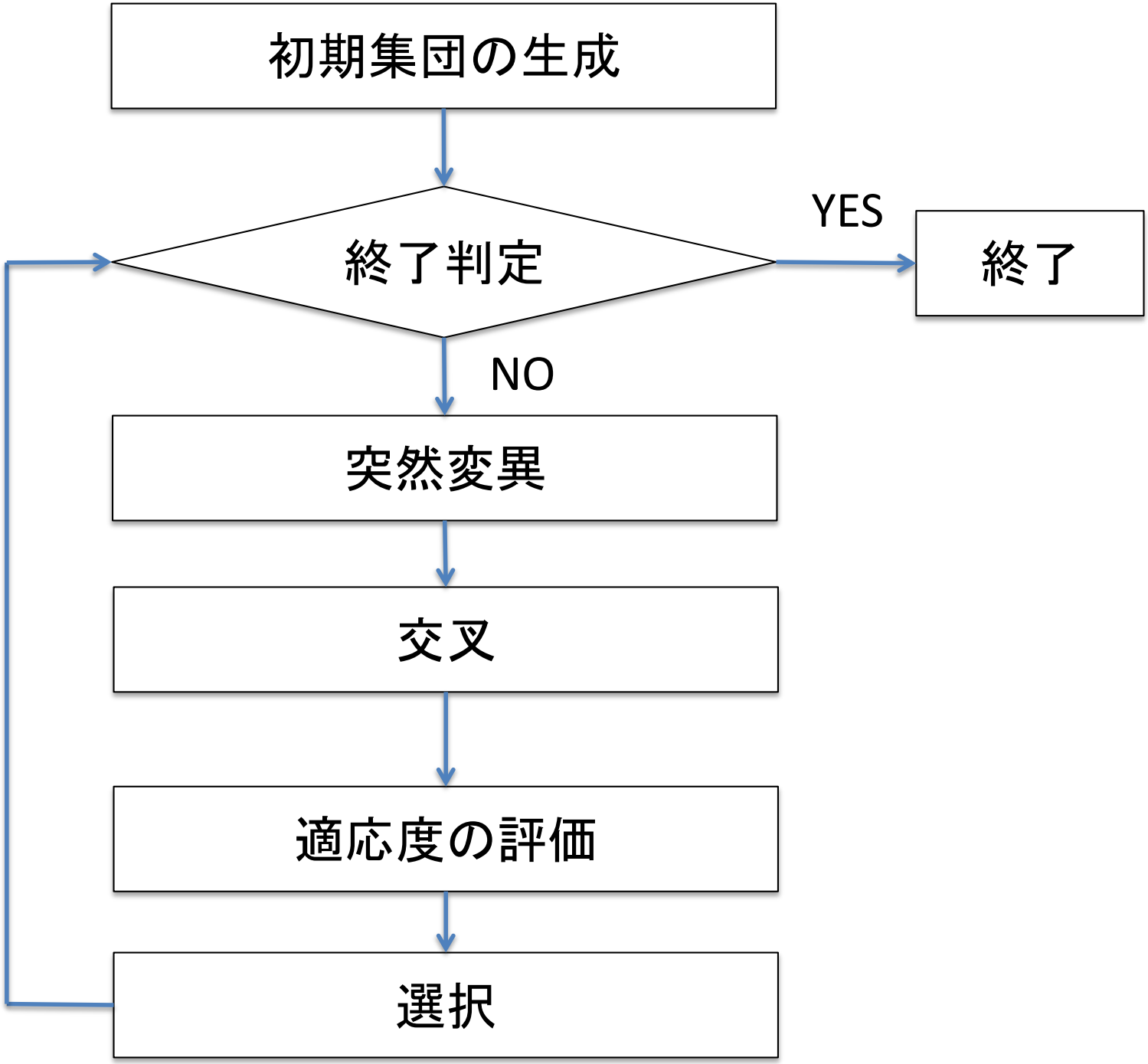
縦軸:各世代の重ね合わせのRMSD

この図で確認すべきポイント

世代数、集団サイズ、突然変異率、組換え率を変更した時、どれだけ早くRMSDが小さくなるか

RMSDが小さい = 構造の重ね合わせが良い

RMSDを減少させるのに、どのパラメータが大きく寄与するかを調べる。


初期集団の生成
終了判定
YES
終了
NO
突然変異
交叉
適応度の評価
選択

[1] 1jl9.pdbと1jl9m.pdbを重ね合わせる

[1-1] 集団サイズ3と集団サイズ100の比較

世代数は100、突然変異率0.7, 組換え率0.7に固定

executeGA.Rのパラメータを変更して実行

同じパラメータで5回実行

~~set.seedの括弧内を回数の値に設定する~~

最終のステップのRMSDの平均

--- 重ね合わせがどれだけうまくいっているかの指標

RMSDがはじめて1.0以下になった時のステップ数の平均

--- どれだけ速く重ね合わせできているかの指標

RMSDの世代数に対するプロット(集団サイズ3と100)、それぞれ5回実施したものの一つ

重ね合わせ前の1jl9.pdbと1j9m.pdbの構造のmolmilによる表示

重ね合わせ後の構造のmolmilによる表示(集団サイズ3と100)

それぞれ5回実施した時のどれか一つについてスクリーンショットを撮る

集団サイズが重ねあわせに及ぼす影響を、上記のデータに基づいて議論

※ 1jl9.pdbと1jl9m.pdbは、同じ構造なので、重ね合わせのRMSDは本来0になるはず

スクリーンショットの撮り方の調べ方

Googleなどで検索

キーワードは　“スクリーンショット”と”Windows”
あるいは
“スクリーンショット”と”Mac”

自分が使っているPCによってWindowsかMacを変えて検索

[1] 1jl9.pdbと1jl9m.pdbを重ね合わせる
[1-2] 世代数5と世代数100の比較
　集団サイズは100、突然変異率0.7, 組換え率0.7に固定
executeGA.Rのパラメータを変更して実行

同じパラメータで5回実行
~~set.seedの括弧内を回数の値に設定する~~

最終のステップのRMSDの平均
　　--- 重ね合わせがどれだけうまくいっているかの指標

RMSDの世代数に対するプロット(世代数5と100)、それぞれ
5回実施したものの一つ

重ね合わせ前の1jl9.pdbと1j9m.pdbの構造のmolmilによる表示
重ね合わせ後の構造のmolmilによる表示(世代数5と世代数100)
それぞれ5回実施した時のどれか一つについてスクリーンショットを撮る

世代数が重ねあわせに及ぼす影響を、上記のデータに基づいて議論
※ 1jl9.pdbと1jl9m.pdbは、同じ構造なので、重ね合わせのRMSDは
本来0になるはず

[1] 1jl9.pdbと1jl9m.pdbを重ね合わせる
[1-3] 突然変異率0と突然変異率0.7の比較
　集団サイズは100、世代数100, 組換え率0.7に固定
executeGA.Rのパラメータを変更して実行
同じパラメータで5回実行
~~set.seedの括弧内を回数の値に設定する~~
　最終のステップのRMSDの平均
　　--- 重ね合わせがどれだけうまくいっているかの指標
　RMSDがはじめて1.0以下になった時のステップ数の平均
　　--- どれだけ速く重ね合わせできているかの指標

　RMSDの世代数に対するプロット(~~世代数5と100, 突然変異率0と0.7~~)、
　それぞれ5回実施したもののの一つ

　重ね合わせ前の1jl9.pdbと1j9m.pdbの構造のmolmilによる表示
　重ね合わせ後の構造のmolmilによる表示(突然変異率0と0.7)
　それぞれ5回実施した時のどれか一つについてスクリーンショットを撮る

突然変異率が重ねあわせに及ぼす影響を、上記のデータに基づいて議論
※ 1jl9,pdbと1jl9m.pdbは、同じ構造なので、重ね合わせのRMSDは本来0になるはず

[1] 1jl9.pdbと1jl9m.pdbを重ね合わせる
[1-4] 組換え率0と組換え率0.7の比較
　集団サイズは100、世代数100, 突然変異率0.7に固定
executeGA.Rのパラメータを変更して実行
同じパラメータで5回実行
~~set.seedの括弧内を回数の値に設定する~~
　最終のステップのRMSDの平均
　　--- 重ね合わせがどれだけうまくいっているかの指標
　RMSDがはじめて1.0以下になった時のステップ数の平均
　　--- どれだけ速く重ね合わせできているかの指標
　　1.0に到達しなければステップ数は100とする

　RMSDの世代数に対するプロット(~~世代数5と100~~組換え率0と0.7)、
　それぞれ5回実施したもののの一つ

　重ね合わせ前の1jl9.pdbと1j9m.pdbの構造のmolmilによる表示
　重ね合わせ後の構造のmolmilによる表示(組換え率0と組換え率0.7)
　それぞれ5回実施した時のどれか一つについてスクリーンショットを撮る

組換え率が重ねあわせに及ぼす影響を、上記のデータに基づいて議論
※ 1jl9,pdbと1jl9m.pdbは、同じ構造なので、重ね合わせのRMSDは
本来0になるはず

[2] 3c9a.pdbと1jl9.pdbを重ね合わせる

[2-1] 集団サイズ3と集団サイズ100の比較

世代数は100、突然変異率0.7, 組換え率0.7に固定

executeGA.Rのパラメータを変更して実行

同じパラメータで5回実行

~~set.seedの括弧内を回数の値に設定する~~

最終のステップのRMSDの平均

--- 重ね合わせがどれだけうまくいっているかの指標

RMSDがはじめて**3.0**以下になった時のステップ数の平均

--- どれだけ速く重ね合わせできているかの指標

RMSDの世代数に対するプロット(集団サイズ3と100)、それぞれ5回実施したもののの一つ

重ね合わせ前の1jl9.pdbと3c9a.pdbの構造のmolmilによる表示

重ね合わせ後の構造のmolmilによる表示(集団サイズ3と100)

それぞれ5回実施した時のどれか一つについてスクリーンショットを撮る

集団サイズが重ねあわせに及ぼす影響を、上記のデータに基づいて議論

[2] 1jl9.pdbと3c9a.pdbを重ね合わせる
[2-2] 世代数5と世代数100の比較
　集団サイズは100、突然変異率0.7, 組換え率0.7に固定
executeGA.Rのパラメータを変更して実行

同じパラメータで5回実行
~~set.seedの括弧内を回数の値に設定する~~

最終のステップのRMSDの平均
　　--- 重ね合わせがどれだけうまくいっているかの指標

RMSDの世代数に対するプロット(世代数5と100)、それぞれ
5回実施したもののの一つ

重ね合わせ前の1jl9.pdbと3c9a.pdbの構造のmolmilによる表示
重ね合わせ後の構造のmolmilによる表示(世代数5と世代数100)
それぞれ5回実施した時のどれか一つについてスクリーンショットを撮る

世代数が重ねあわせに及ぼす影響を、上記のデータに基づいて議論

[2] 1jl9.pdbと3c9a.pdbを重ね合わせる
[2-3] 突然変異率0と突然変異率0.7の比較
　集団サイズは100、世代数100, 組換え率0.7に固定
executeGA.Rのパラメータを変更して実行
同じパラメータで5回実行
~~set.seedの括弧内を回数の値に設定する~~
　最終のステップのRMSDの平均
　　--- 重ね合わせがどれだけうまくいっているかの指標
　RMSDがはじめて**3.0**以下になった時のステップ数の平均
　　--- どれだけ速く重ね合わせできているかの指標

　RMSDの世代数に対するプロット(~~世代数5と100~~ 突然変異率0と0.7)、
　それぞれ5回実施したもののの一つ

　重ね合わせ前の1jl9.pdbと3c9a.pdbの構造のmolmilによる表示
　重ね合わせ後の構造のmolmilによる表示(突然変異率0と0.7)
　それぞれ5回実施した時のどれか一つについてスクリーンショットを撮る

突然変異率が重ねあわせに及ぼす影響を、上記のデータに基づいて議論

[2] 1jl9.pdbと3c9a.pdbを重ね合わせる
[2-4]組換え率0と組換え率0.7の比較
　集団サイズは100、世代数100， 突然変異率0.7に固定
executeGA.Rのパラメータを変更して実行
同じパラメータで5回実行
~~set.seedの括弧内を回数の値に設定する~~
　最終のステップのRMSDの平均
　　--- 重ね合わせがどれだけうまくいっているかの指標
　RMSDがはじめて3.0以下になった時のステップ数の平均
　　--- どれだけ速く重ね合わせできているかの指標
　　3.0に到達しなければステップ数は100とする

　RMSDの世代数に対するプロット(~~世代数5と100~~ 組換え率0と0.7)、それぞれ5回実施したものの一つ

　重ね合わせ前の1jl9.pdbと3c9a.pdbの構造のmolmilによる表示
　重ね合わせ後の構造のmolmilによる表示(組換え率0と組換え率0.7)
　それぞれ5回実施した時のどれか一つについてスクリーンショットを撮る

組換え率が重ねあわせに及ぼす影響を、上記のデータに基づいて議論

[3] 3c9a.pdbと1jl9.pdbを重ね合わせる

[3-1]集団サイズは500、世代数100, 突然変異率0.7, 組換え率0.7で重ね合わせを実施

~~5回set.seedの値を変えながら実行~~

同じパラメータで5回実行

最終のステップのRMSDの平均

--- 重ね合わせがどれだけうまくいっているかの指標

RMSDがはじめて**3.0**以下になった時のステップ数の平均

--- どれだけ速く重ね合わせできているかの指標

3.0に到達しなければステップ数は100とする

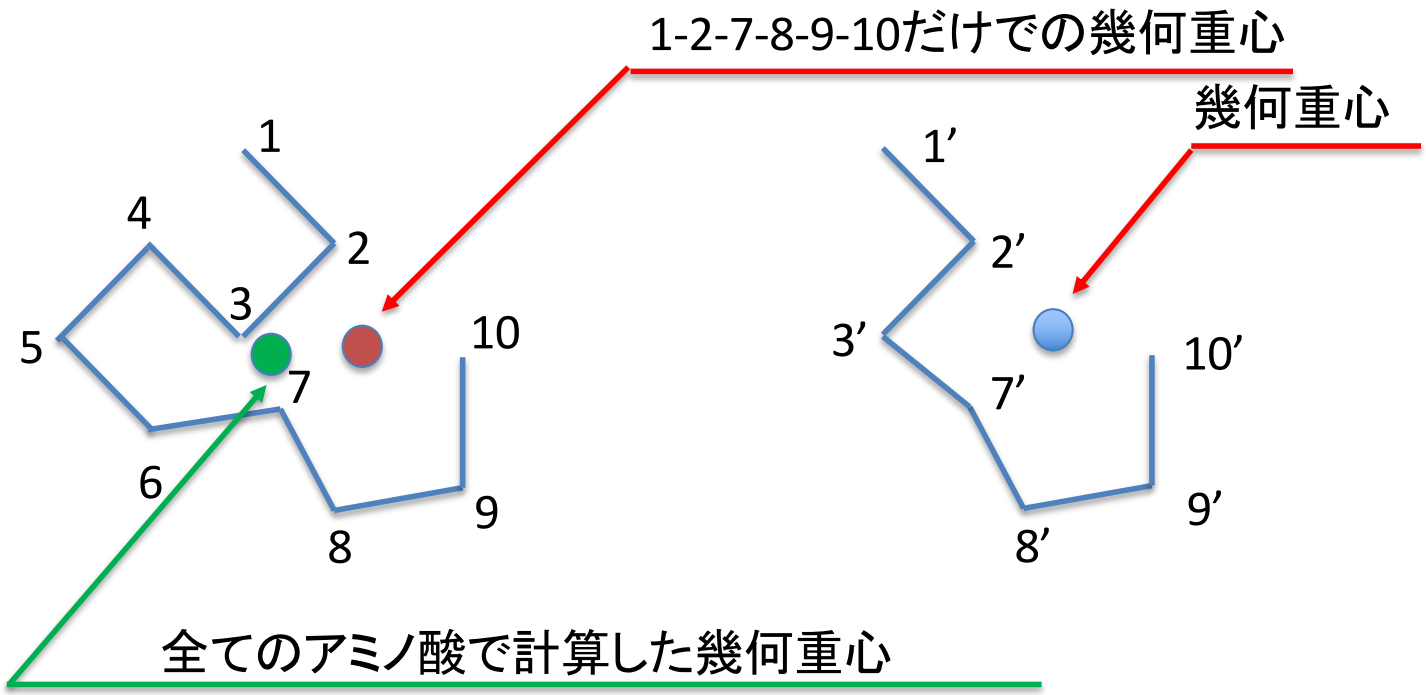
重ね合わせ前の3c9a.pdbと1j9.pdbの構造のmolmilによる表示

重ね合わせ後の構造のmolmilによる表示

それぞれ5回実施した時のどれか一つについてスクリーンショットを撮る

初期配置に比べると重なっているが、ズレが大きいことを確認

----→ プログラムを書き換えて重ね合わせを改善する。



アラインメント

1	2	3	4	5	6	7	8	9	10
1′	2′	3′	–	–	–	7′	8′	9′	10′

重心あわせの後、アラインメント上対応する残基で回転させてRMSDを最小化
現在のプログラムでは全てのアミノ酸のCaで幾何重心を計算
アミノ酸4-5-6 (挿入）のせいで対応するアミノ酸の重心の位置がずれる

重心計算の際、アラインメントで対応するアミノ酸のCαだけを使えば
より重ね合わせが良くなる？

[3] 3c9a.pdbと1jl9.pdbを重ね合わせる
[3-2] FunctionLibraryY.Rの該当箇所を変更して、[3-1]と同じパラメータ値でexecuteGA.Rを実行

同じパラメータで5回実行
最終のステップのRMSDの平均
--- 重ね合わせがどれだけうまくいっているかの指標
RMSDがはじめて3.0以下になった時のステップ数の平均
--- どれだけ速く重ね合わせできているかの指標
3.0に到達しなければステップ数は100とする

重ね合わせ前の3c9a9.pdbと1j9.pdbの構造のmolmilによる表示
重ね合わせ後の構造のmolmilによる表示
それぞれ5回実施した時のどれか一つについてスクリーンショットを撮る

[3-1]の結果に比べて重ね合わせが改善されていることを確認

FunctionLibraryY.Rの

#1つ目のPDBのファイルの幾何重心を計算　(77行 – 79行) をコメントアウト

```
ca1gx <- mean(ca1$x)
ca1gy <- mean(ca1$y)
ca1gz <- mean(ca1$z)
```

#2つ目のPDBファイルの幾何重心を計算　(120行 – 122行) をコメントアウト

```
ca2gx <- mean(ca2$x)
ca2gy <- mean(ca2$y)
ca2gz <- mean(ca2$z)
```

ca1gx, ca1gy, ca1gz, ca2gx, ca2gy, ca2gzの計算に
おいて、変数alignから対応する残基のみを取り出し、その残基だけで
重心を計算する

※ コメントアウトとは、コマンド行をコメントとして実行から外すこと
上の例では、行頭に#を入れると良い。例えば77行の場合

```
#ca1gx <- mean(ca1$x)
```

ヒント：36行でアラインメントのFASTAファイルを読み込んでいる変数alignを使う

```
align$ali
                            [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10] [,11] [,12] [,13] [,14] [,15] [,16] [,17] [,18]
3C9A:C|PDBID|CHAIN|SEQUENCE "P"  "T"  "Y"  "K"  "C"  "P"  "E"  "T"  "F"  "D"   "A"   "W"   "Y"   "C"   "L"   "N"   "D"   "A"
1JL9B:EPIDERMAL             "-"  "-"  "-"  "-"  "C"  "P"  "L"  "S"  "H"  "D"   "G"   "-"   "Y"   "C"   "L"   "H"   "D"   "G"
                            [,19] [,20] [,21] [,22] [,23] [,24] [,25] [,26] [,27] [,28] [,29] [,30] [,31] [,32] [,33] [,34]
3C9A:C|PDBID|CHAIN|SEQUENCE "H"   "C"   "F"   "A"   "V"   "K"   "I"   "A"   "D"   "L"   "P"   "V"   "Y"   "S"   "C"   "E"
1JL9B:EPIDERMAL             "V"   "C"   "M"   "Y"   "I"   "E"   "A"   "L"   "D"   "-"   "-"   "K"   "Y"   "A"   "C"   "N"
                            [,35] [,36] [,37] [,38] [,39] [,40] [,41] [,42] [,43] [,44] [,45] [,46] [,47] [,48] [,49]
3C9A:C|PDBID|CHAIN|SEQUENCE "C"   "A"   "I"   "G"   "F"   "M"   "G"   "Q"   "R"   "C"   "E"   "Y"   "K"   "E"   "-"
1JL9B:EPIDERMAL             "C"   "V"   "V"   "G"   "Y"   "I"   "G"   "E"   "R"   "C"   "Q"   "Y"   "R"   "D"   "L"
```

align$ali[1,]に3c9aのアラインメントが得られる
align$ali[2,]に1jl9のアラインメントが得られる
align$ali[1,4]はK、align$ali[2,4]は'-'を

3c9aの重心を求める際、赤字の残基は、1jl9に相手がいないので(gapなので)、重心の計算に使わない
1jl9の重心を求める際、青字の残基は、3c9aに相手がいないので(gapなので)、重心の計算に使わない

アラインメントの長さは　length(align$ali[1,]) あるいは length(align$ali[2,]) で得られる
length(align$ali[1,]) も length(align$ali[2,])も、同じ値

ヒントの続き

立体構造のCaの座標データを含む行は56行、57行で取り出されている

```
ca1 <<- p1$atom[p1$atom[,3]=='CA',]
ca2 <<- p2$atom[p2$atom[,3]=='CA',]
```

立体構造の1番目のアミノ酸のx、y、z座標は `ca1[1,]$x, ca1[1,]$y, ca1[1,]$z` で取り出せる。」

例えば、3c9aの重心を求める時　(81行から97行の先頭の#を削除してコメント行から復帰させる)
アラインメントのサイト番号は、立体構造の残基番号と対応していないことに注意
変数 `siteNo` を用意しておく．　`siteNo <- 0`
空ベクトル `ax1, ay1, az1`を用意しておく。 例: `ax1 <- c()`
for 文で アラインメントの一番目のサイトから最後のサイトまで順番に見ていく
　(1) if文を使う。　3c9aが'-'でない時、`siteNo <- siteNo + 1`
　(2) 相手側(1jl9)のアラインメントサイトが'-'でない時だけ、x, y, zの座標値を重心の計算に利用
　　←- if文を使う

```
siteNo <- 0
ax1 <- c()
ay1 <- c()
az1 <- c()
for (i in  ????)  {
    if (3c9aのi番目のアラインメントサイトが'-'でない場合)  {
       siteNo <- siteNo + 1
       if (1jl9のi番目のアラインメントサイトが'-'でない場合) {
            ax1 <- c(ax1, ca1のsiteNo番目のx座標)
            ay1 <- c(ay1, ca1のsiteNo番目のy座標)
            az1 <- c(az1, ca1のsiteNo番目のz座標)
        }
     }
}
```

ヒントの続き

得られたax1, ay1, az1を使って重心を求める

```
ca1gx <- mean(ax1)
ca1gy <- mean(ay1)
ca1gz <- mean(az1)
```

1jl9についても同様の操作を行う。
コメントになっている106行から122行を復帰させて実施

```
siteNo <- 0
ax2 <- c()
ay2 <- c()
az2 <- c()
for (i in ?????)  {
    if (1jl9のi番目のアラインメントサイトが'-'でない場合)  {
         siteNo <- siteNo + 1
         if (3c9aのi番目のアラインメントサイトが'-'でない場合) {
               ax2 <- c(ax2, ca2のsiteNo番目のx座標)
               ay2 <- c(ay2, ca2のsiteNo番目のy座標)
               az2 <- c(az2, ca2のsiteNo番目のz座標)
         }
      }
}
ca2gx <- mean(ax2)
ca2gy <- mean(ay2)
ca2gz <- mean(az2)
```

6. レポートの構成

レポートの構成

1. 背景
2. 方法
3. 結果
4. 考察

レポートの作成

1.Intoroduction（背景）

（1）立体構造の重ね合わせの説明

（2） 遺伝的アルゴリズムの一般論の説明

（3） Rの説明

（4） 目的

（4-1） 立体構造の重ね合わせを題材として遺伝的アルゴリズムの各種パラメータの及ぼす影響を調べる

（4-2） 重心計算の部分を改良して、重ね合わせを改善する

2．方法

(1) 立体構造の重ねあわせのためのGAの設定の説明

(1-1) 染色体や遺伝子の構成

(1-2) 初期集団はどのように生成したか

(1-3) 突然変異や組換えはどのように行われるか

(1-4) 適応度はどのように定義されているか

(1-5) 選択はどのように行われているか

(1-6) 終了条件

(2) 準備されたExecuteGA.Rと~~FunctionLibraryY.Rと FunctionLibraryX.Rについて~~説明

FunctionLibraryY.RとFunctionLibraryX.Rの中の各関数と(1)の処理との対応

(3) 各種パラメータを変更してGAを実施し、パラメータのGAに及ぼす影響の説明

(4) 現在のスクリプトの重心計算の問題点と改善の方法の説明

3．結果

(1)1jl9.pdbと1jl9m.pdbの重ね合わせ

[1-1] ~ [1-4] (スライドのp197 - 201)の結果を図を含めて説明

(2) 3c9a.pdbと1jl9.pdb の重ね合わせ

[2-1] ~ [2-4] (スライドのp.202 ~ p.205)の結果を図を含めて説明

(3)プログラムを修正して3c9a.pdbと1jl9.pdbを重ね合わせ

[3-1] ~ [3-2] (スライドのp.215 ~ p.221)の結果を図を含めて説明

4. 考察

(1) 3の結果の(1)(2)に基づき、各種パラメータのGAのパフォーマンスに及ぼす影響を議論
何故そうなるのか？

(2) 3の結果の(3)に基づき、重心計算の変更が重ね合わせに及ぼす影響や理由を議論