



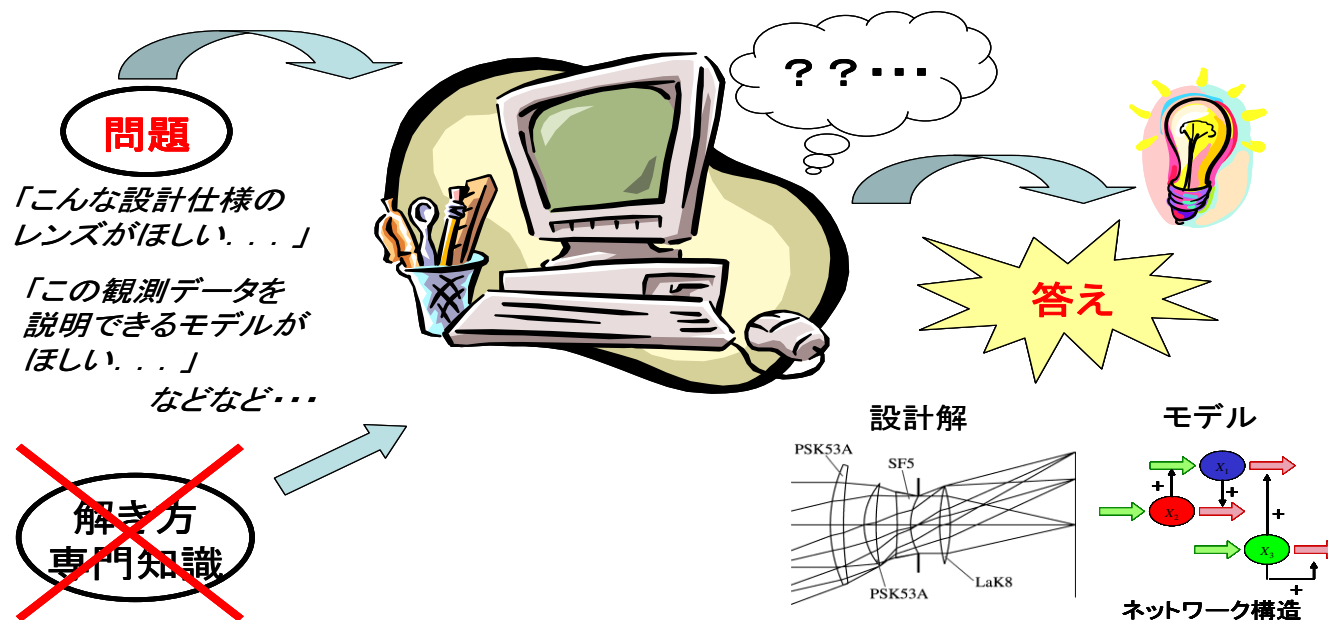
# 進化計算入門

---

小野(功)研究室

# 進化計算とは？

- 生物の進化過程を模倣した人工知能技術の1つ
  - 何が嬉しいのか？・・・教師データが不要. 計算機自らが試行錯誤.
  - 人間を超える設計解／モデルの発見
  - 無駄な試行錯誤はしたくない=>進化計算の研究



# 身近なところで使われている 進化計算(1)

- N700系新幹線の先頭形状の設計



列島宝物館 (<http://www.i-treasury.net/>)

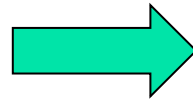
# 身近なところで使われている 進化計算(1)

- N700系新幹線の先頭形状の設計

0系



列島宝物館 (<http://www.i-treasury.net/>)



100系



列島宝物館 (<http://www.i-treasury.net/>)

**より滑らかに！より細長く！**

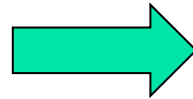
# 身近なところで使われている 進化計算(1)

- N700系新幹線の先頭形状の設計

100系



列島宝物館 (<http://www.i-treasury.net/>)



300系



列島宝物館 (<http://www.i-treasury.net/>)

**より滑らかに！より細長く！**



# 身近なところで使われている 進化計算(1)

- N700系新幹線の先頭形状の設計

300系



列島宝物館 (<http://www.i-treasury.net/>)



500系



列島宝物館 (<http://www.i-treasury.net/>)

より滑らかに！ より細長く！

➡ なぜか？

# 身近なところで使われている 進化計算(1)

- N700系新幹線の先頭形状の設計

300系



列島宝物館 (<http://www.i-treasury.net/>)

500系



[https://upload.wikimedia.org/wikipedia/commons/2/2e/500kei\\_himeji.jpg](https://upload.wikimedia.org/wikipedia/commons/2/2e/500kei_himeji.jpg)

しかし、ここで大問題が...

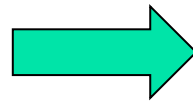
# 身近なところで使われている 進化計算(1)

- N700系新幹線の先頭形状の設計

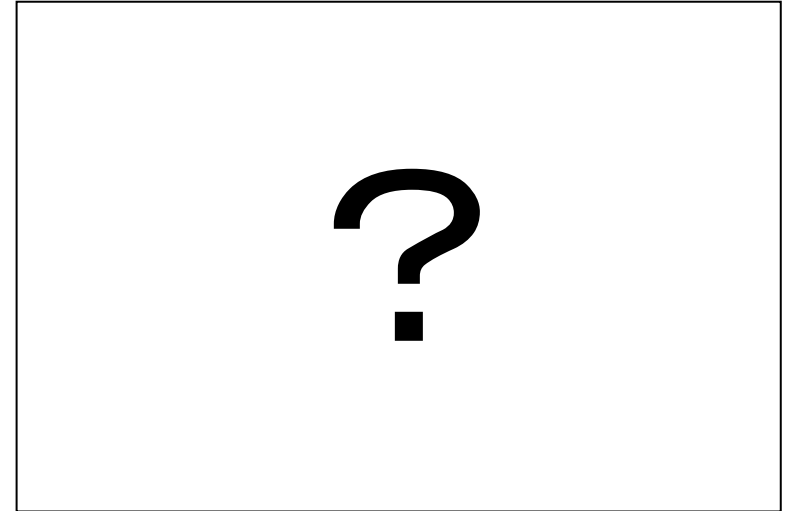
500系



列島宝物館 (<http://www.i-treasury.net/>)



N700系



先頭形状をもっと細長くできない！  
ドア数も座席数も変えられない！ → どうしよう…



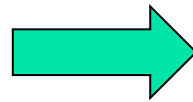
# 身近なところで使われている 進化計算(1)

- N700系新幹線の先頭形状の設計

500系



列島宝物館 (<http://www.i-treasury.net/>)



N700系



列島宝物館 (<http://www.i-treasury.net/>)

進化計算で試行錯誤

# 身近なところで使われている 進化計算(2)

- 旅客機の設計
  - 国産旅客機MRJ
  - 翼の形状



[https://upload.wikimedia.org/wikipedia/commons/7/79/MRJ\\_First\\_Flight\\_%282%29\\_%28cropped%29.png?uselang=ja](https://upload.wikimedia.org/wikipedia/commons/7/79/MRJ_First_Flight_%282%29_%28cropped%29.png?uselang=ja)

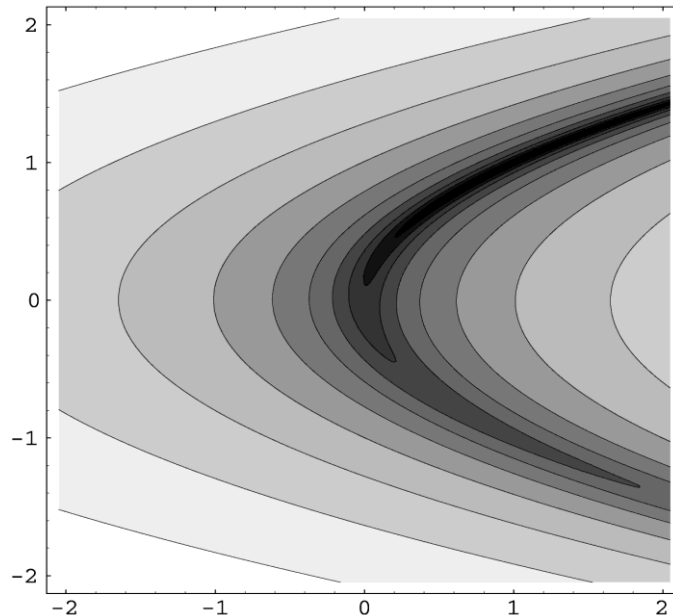
- タイヤの設計
  - 材質
  - 形状



[http://pngimg.com/uploads/tire/tire\\_PNG46.png](http://pngimg.com/uploads/tire/tire_PNG46.png)

# 関数最適化のためのGA(1)

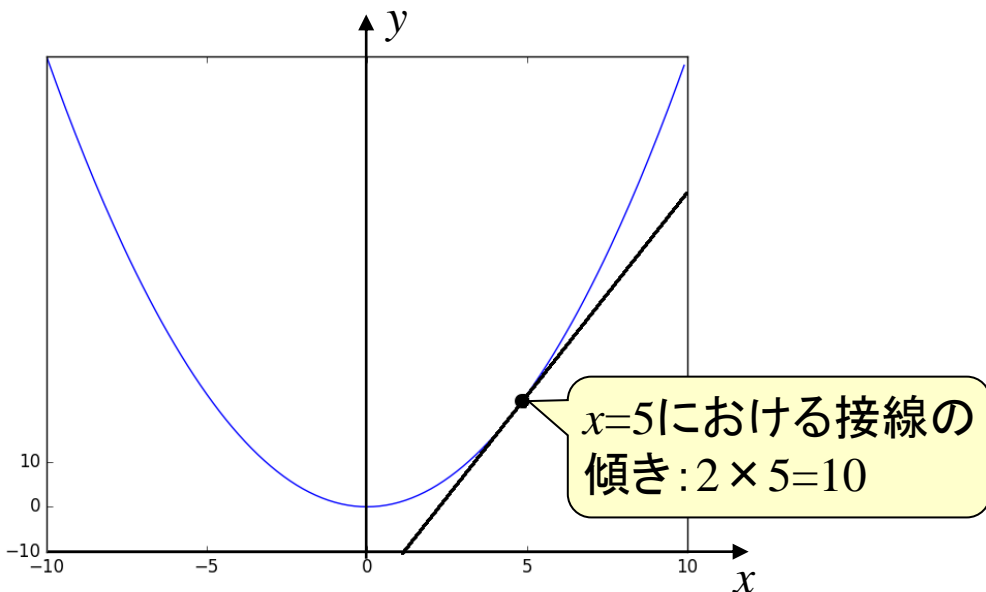
- デモンストレーション
  - 2次元Rosenbrock関数
    - 放物線状の谷
    - $(1, 1)$ で最小値0.0をとる



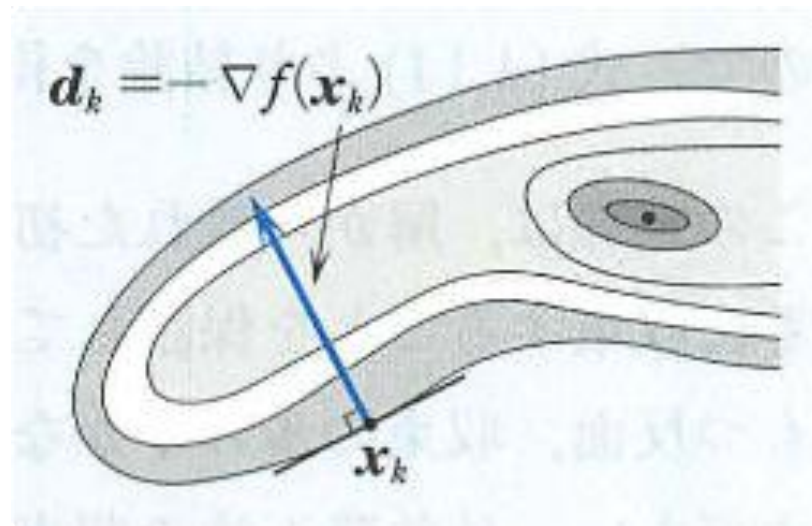
# 関数最適化のためのGA(2)

- ブラックボックス関数最適化
  - 式の形が陽に与えられない
  - 微分(各点における接線の傾き, 勾配)を計算できない

$$y = x^2 \text{ の微分: } \frac{dy}{dx} = 2x$$



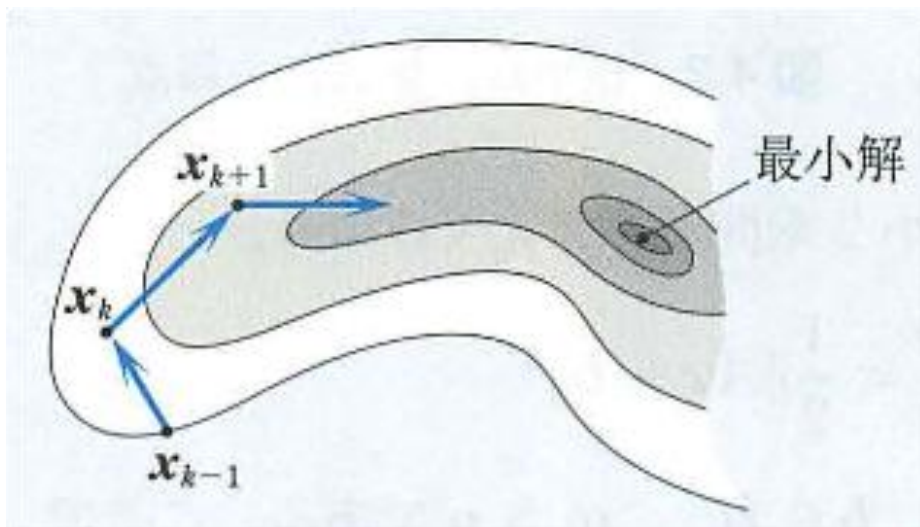
2次元では等高線の法線





# 関数最適化のためのGA(2)

- ブラックボックス関数最適化
  - 式の形が陽に与えられない
  - 微分(各点における接線の傾き, 勾配)を計算できない



勾配方向へ少しずつ  
進んでいけば  
最小解へ辿り着く！

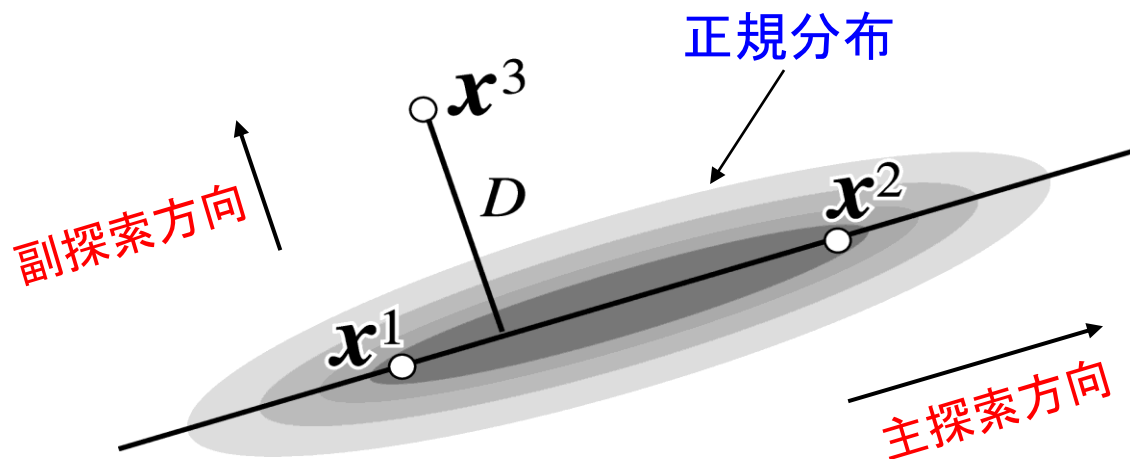
# 関数最適化のためのGA(3)

- 単峰性正規分布交叉(UNDX)

- 交叉

- 単峰性正規分布交叉(Unimodal Normal Distribution Crossover; UNDX)

- (例) 2変数関数最適化の場合



平均ベクトル(中心)

$$m = \frac{x_1 + x_2}{2}$$

主探索方向の標準偏差(広がり)

$$\sigma_{\xi} = 0.5 \|x_1 - x_2\|$$

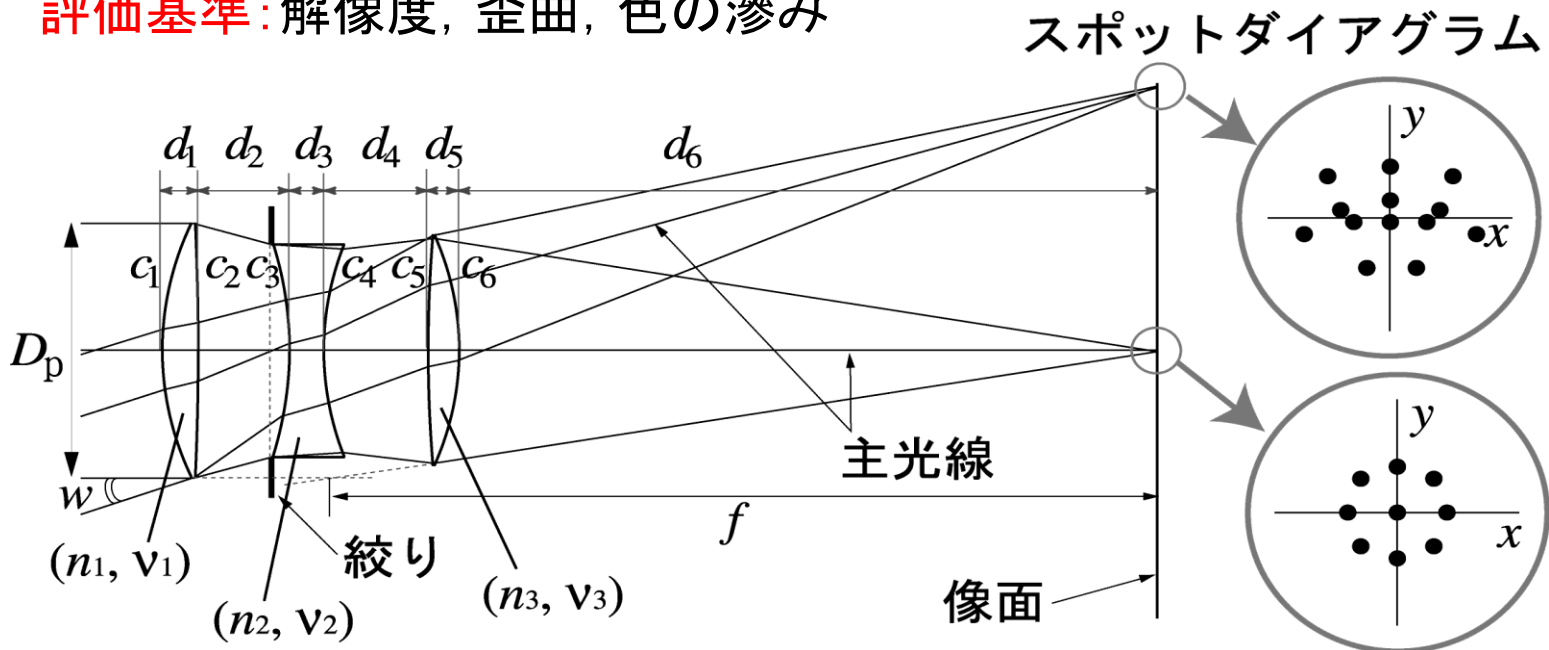
副探索方向の標準偏差(広がり)

$$\sigma_{\eta} = 0.35 D / \sqrt{n}$$

# レンズ設計(1)

## ■ レンズ設計問題

- **設計仕様**: 焦点距離, 明るさ, 画角, バックフォーカス, etc.
- **決定変数**: 曲率, 面間隔, ガラスの材質
- **評価基準**: 解像度, 歪曲, 色の滲み

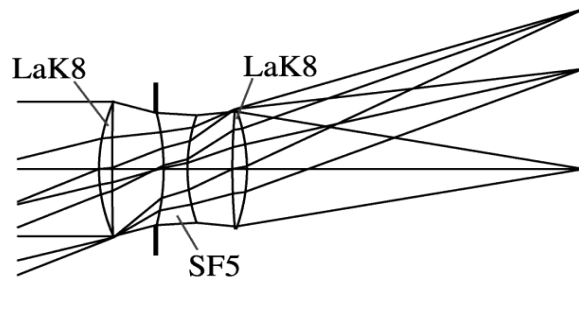


➡ 「専門家の知識を用いない自動設計手法の実現は無理」と言われている

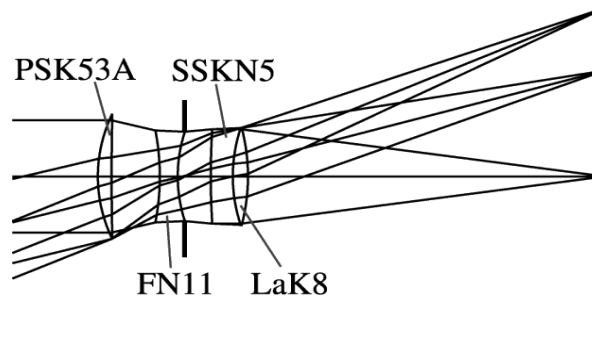
# レンズ設計(2)

## ■ UNDX+MGGによる設計例

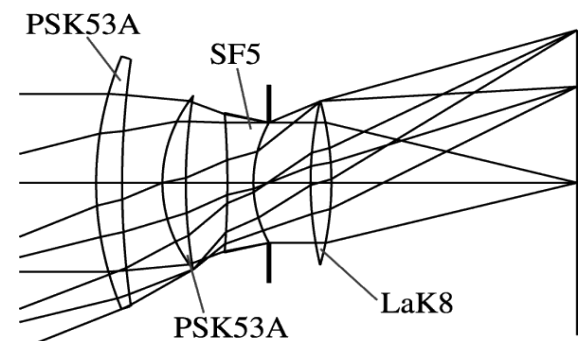
### 小規模レンズ系の設計例



トリプレット型

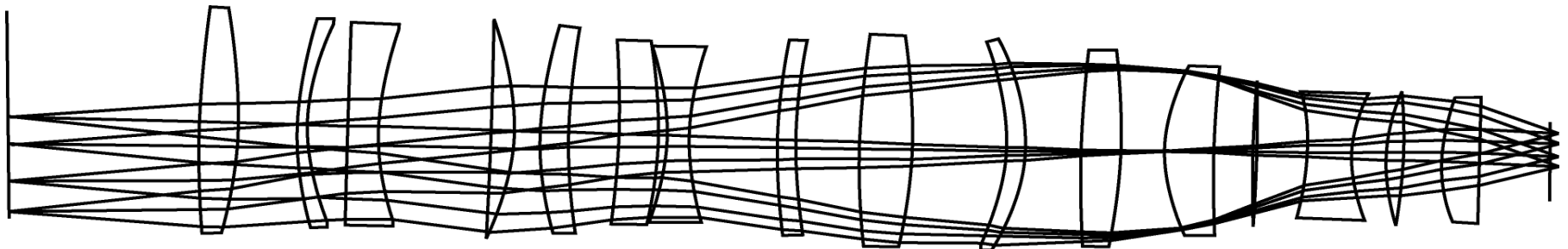


テッサー型



エルノスター型

### 大規模レンズ系の設計例



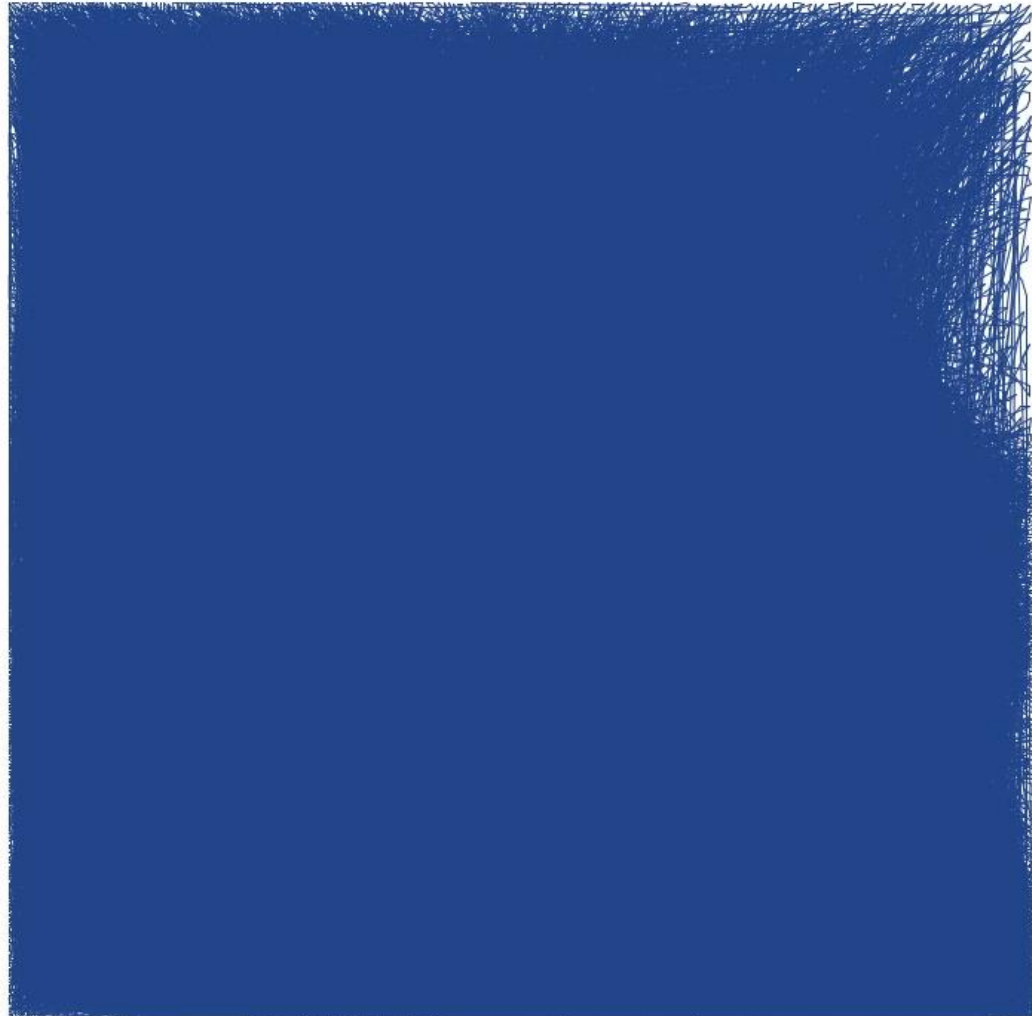




# 大規模組合せ最適化のための 遺伝アルゴリズム

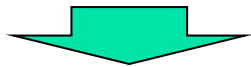
---

- これは何？



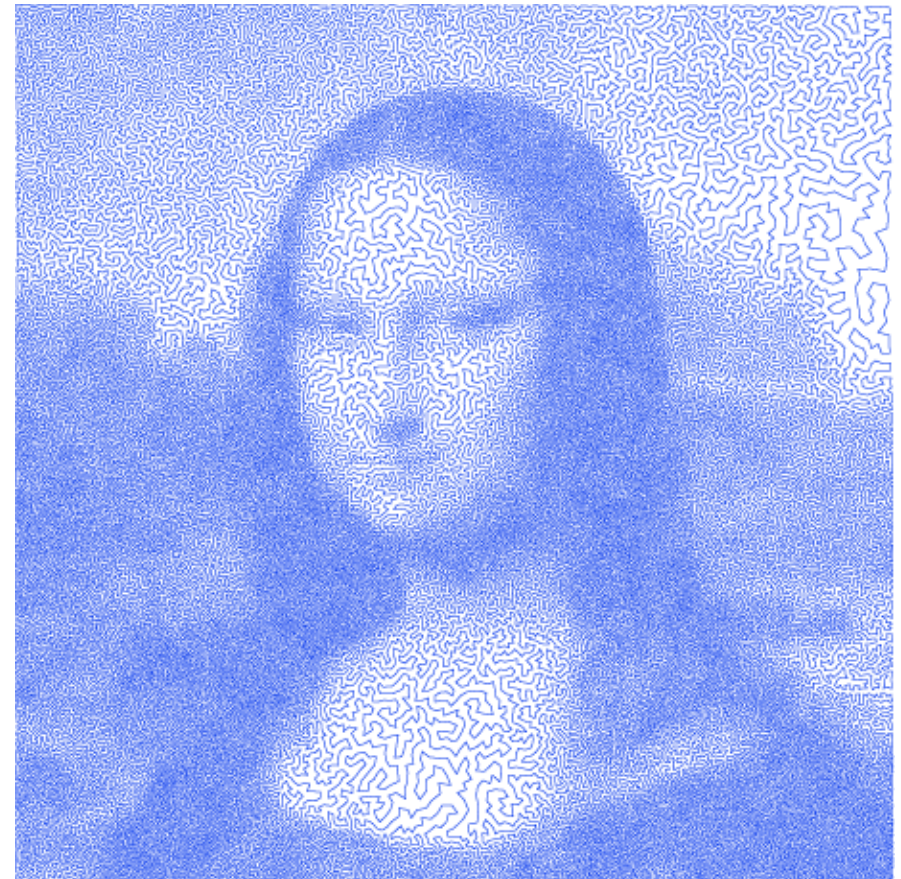
# 大規模組合せ最適化のための 遺伝アルゴリズム

- 巡回セールスマン問題(TSP)
  - 都市数に対して階乗のオーダーで解候補数が爆発
  - 数え上げ戦略では, 30都市程度で, 一生かかっても計算が終わらない
  - Mona-Lisa TSP . . . 10万都市
    - 10万点からなるMona-Lisaの点描の一筆書きの経路長最小化
  - どのように解く？



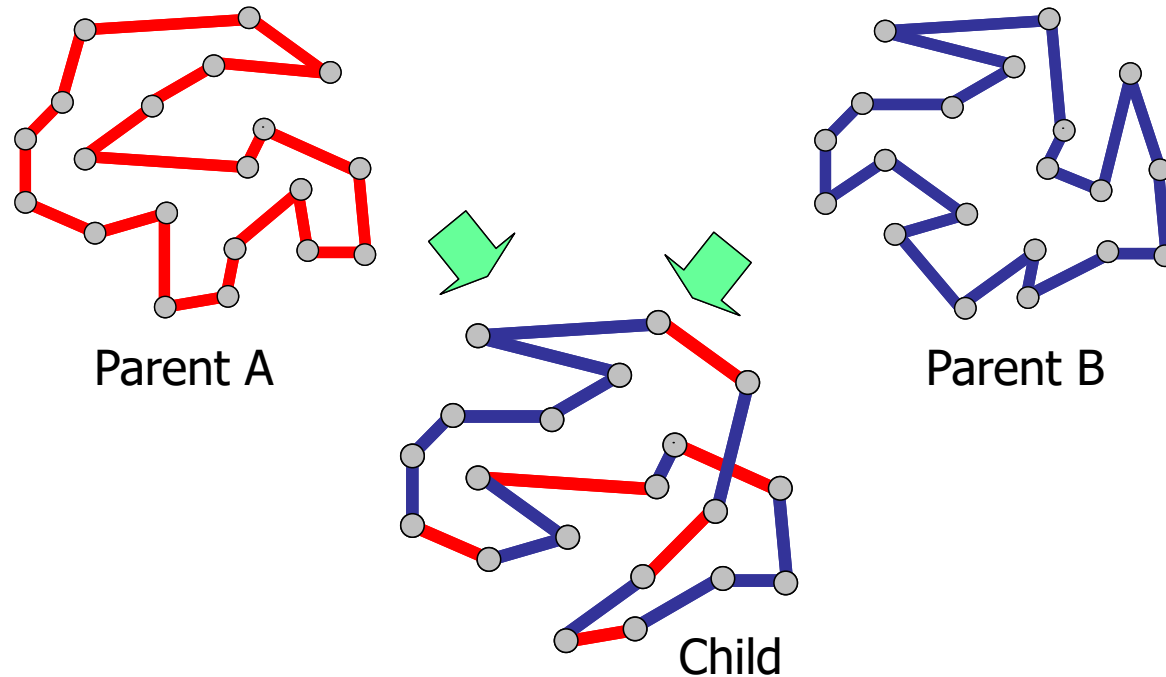
遺伝アルゴリズム  
並列GA-EAX [Honda 13]

**Mona-Lisa TSP**



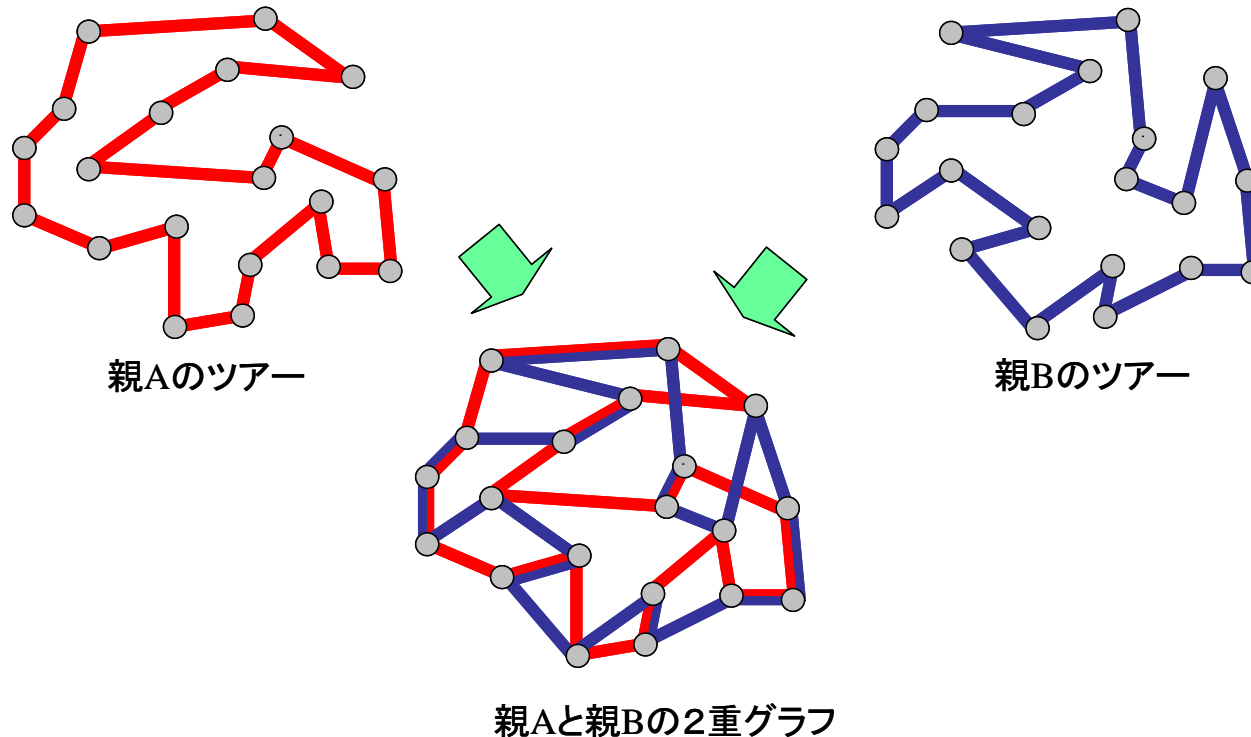
# 大規模組合せ最適化のための並列GA(2)

- 子の生成方法・・・EAX (Edge Assembly Crossover) [Nagata 97]
  - 親 $p_A$ と親 $p_B$ のエッジを受け継ぐように子を生成
    - 淘汰で生き残った親のエッジは良いエッジのはず！



# 大規模組合せ最適化のための並列GA(2)

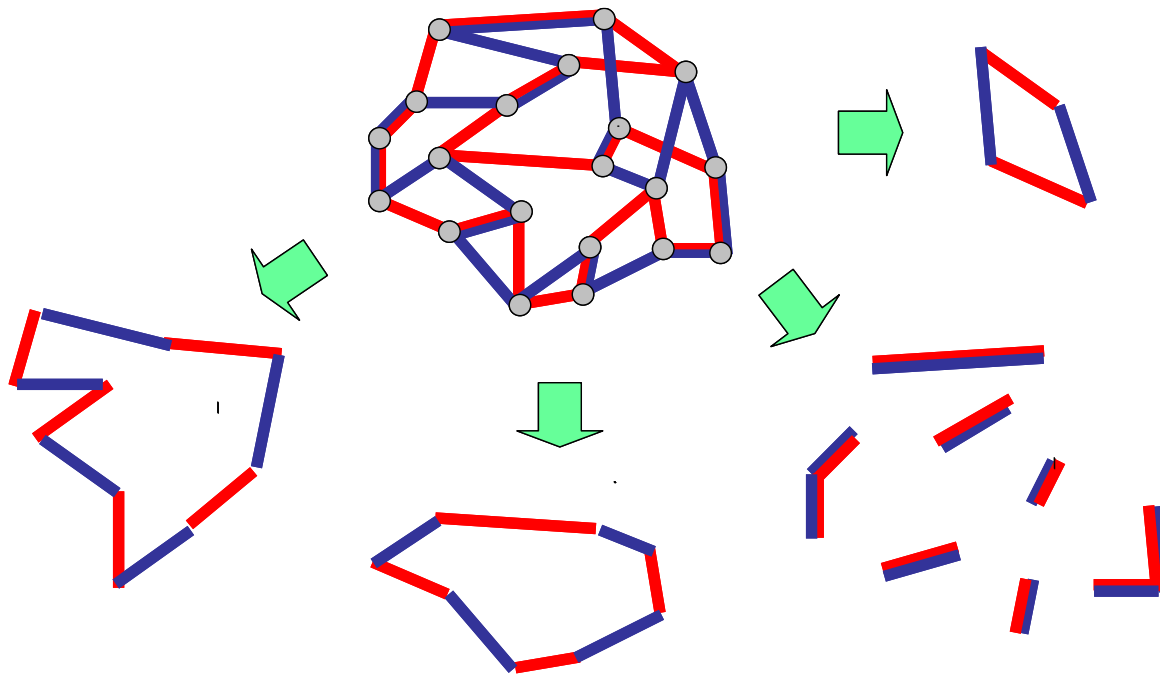
- EAX (Edge Assembly Crossover) [Nagata 97](1)
  - 手順1: 2重グラフの作成





# 大規模組合せ最適化のための並列GA(2)

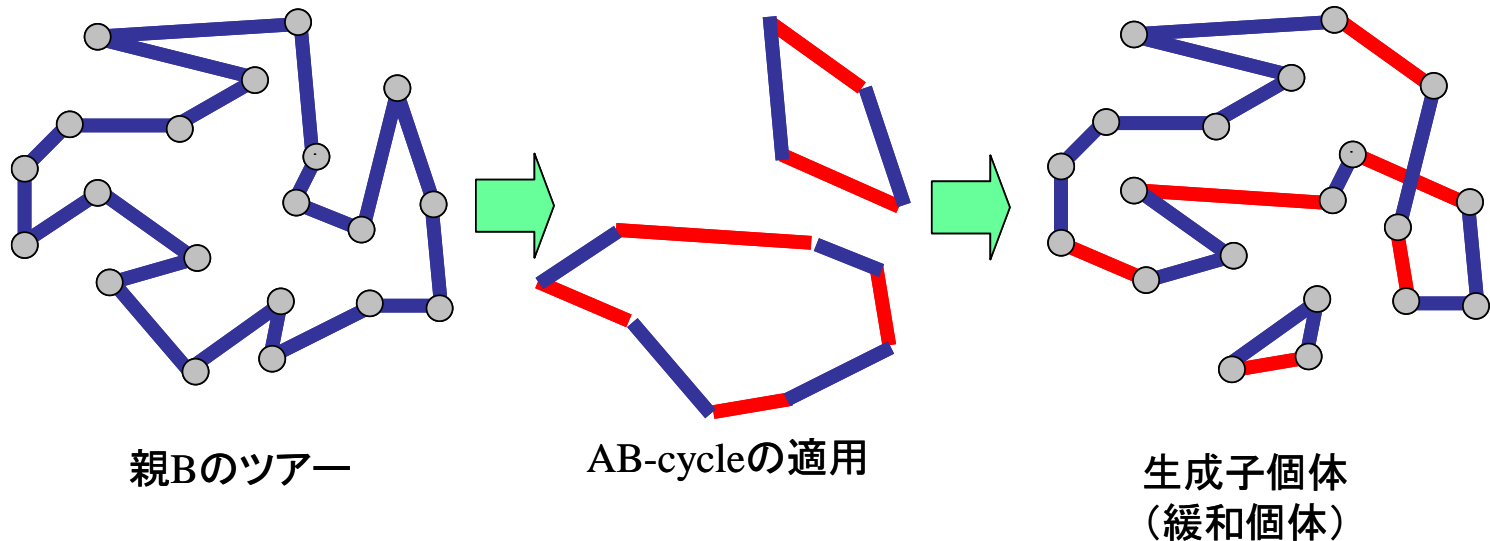
- EAX (Edge Assembly Crossover) [Nagata 97](2)
  - 手順2: AB-cycle集合への分解



■ 2重グラフはAB-cycleの集合に分解できる

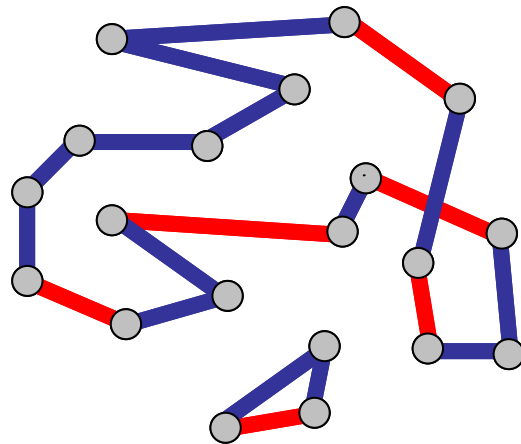
# 大規模組合せ最適化のための並列GA(2)

- EAX (Edge Assembly Crossover) [Nagata 97](3)
  - 手順3: AB-cycleの適用

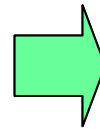


# 大規模組合せ最適化のための並列GA(2)

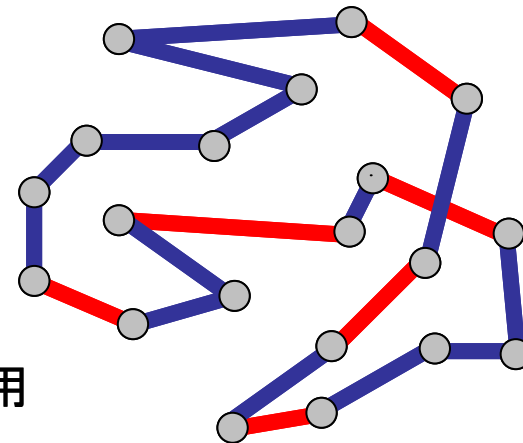
- EAX (Edge Assembly Crossover) [Nagata 97](4)
  - 手順4: 2-optによる修正



緩和個体



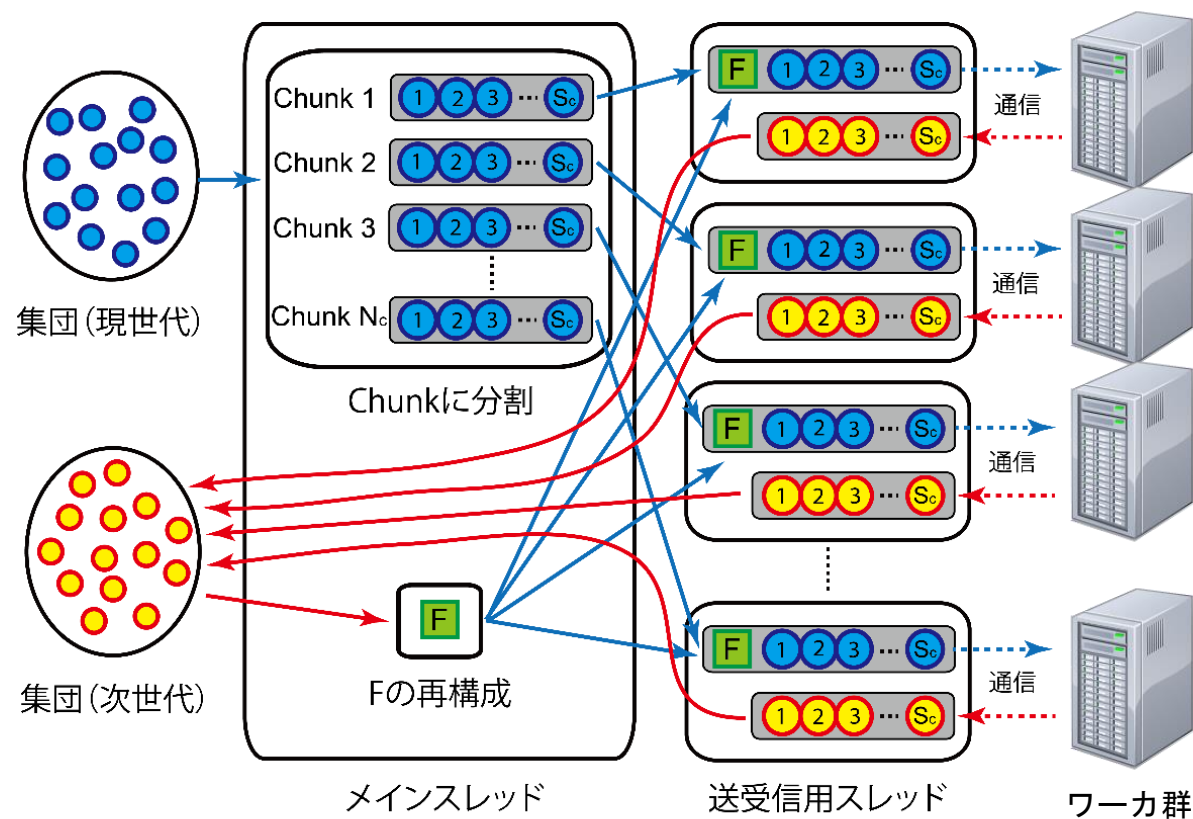
2-optの適用



完全個体

# 大規模組合せ最適化のための並列GA(3)

## ■ 並列GA

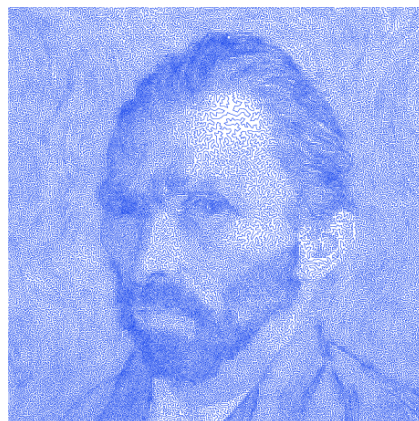




# 大規模組合せ最適化のための並列GA(4)



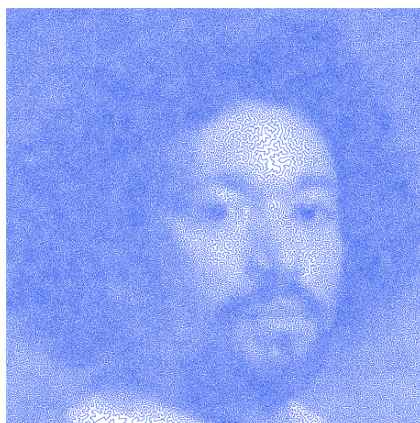
Mona-Lisa100K (10万都市)



Vangogh120K (12万都市)



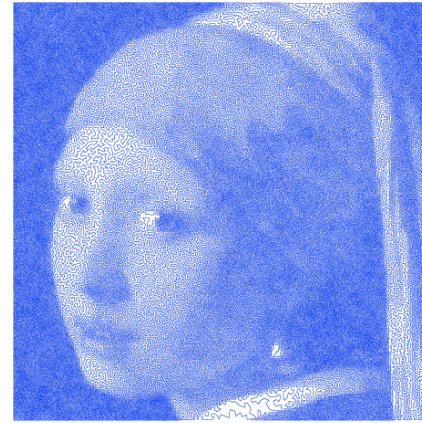
Venus140K (14万都市)



Pareja160K (16万都市)



Courbet180K (18万都市)



Earring200K (20万都市)



## 大規模組合せ最適化のための並列GA(5)

- TSUBAME2.0のノード30台, 約1日の計算時間.
- 10~20万都市のArtTSP  
=> **2つの問題で世界記録を更新！！**

Instance	Best-Known	Proposed GA
mona-lisa100K	5757191	5757191
vangogh120K	6543610	<b>6543609</b>
venus140K	6810665	6810665
pareja160K	7619953	7619953
courbet180K	7888733	<b>7888731</b>
earring200K	8171677	8171677



# おわりに

---

## ■ 進化計算

- 計算機自らが試行錯誤を行うことにより問題解決
- ○What to do? ×How to do?

## ■ モデル化/シミュレーション

＋進化計算＋並列計算

＝新たな問題解決の方法論

- 熟練エンジニアのための設計支援システム
  - レンズ, 新幹線, 飛行機, タイヤなど
- 研究者のためのモデリング／分析支援システム
  - 蛋白質立体構造解析, 遺伝子ネットワーク解析など



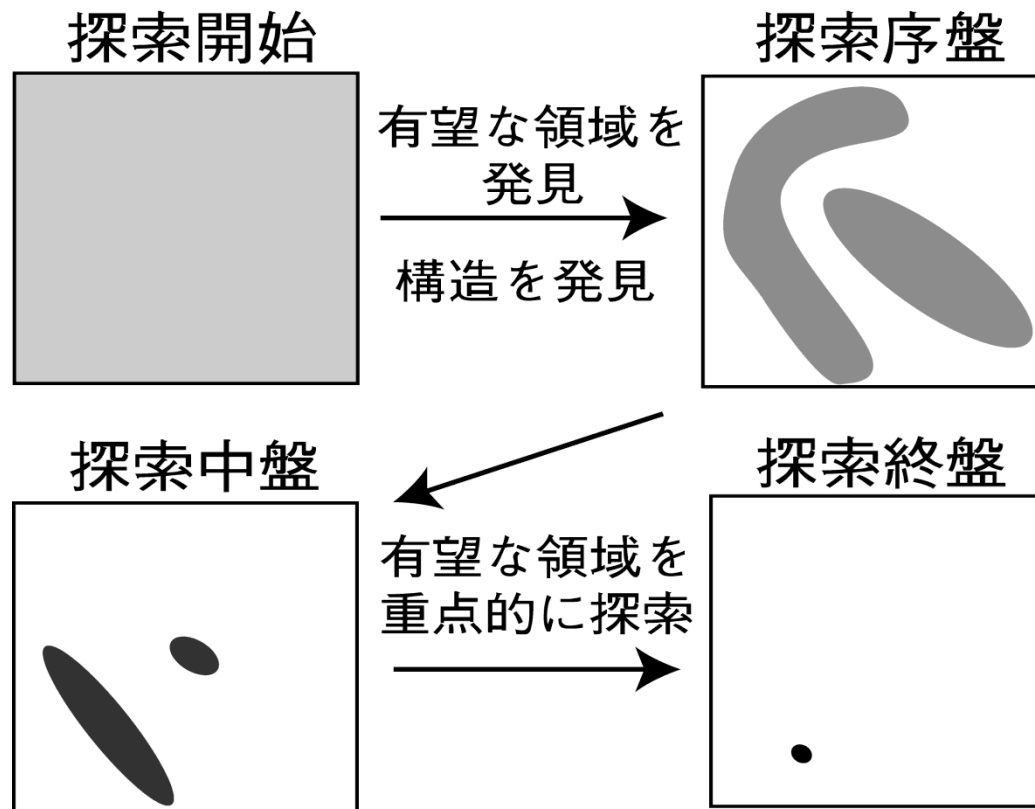
# 課題編

---

# 設計指針(1)

- 探索のシナリオ

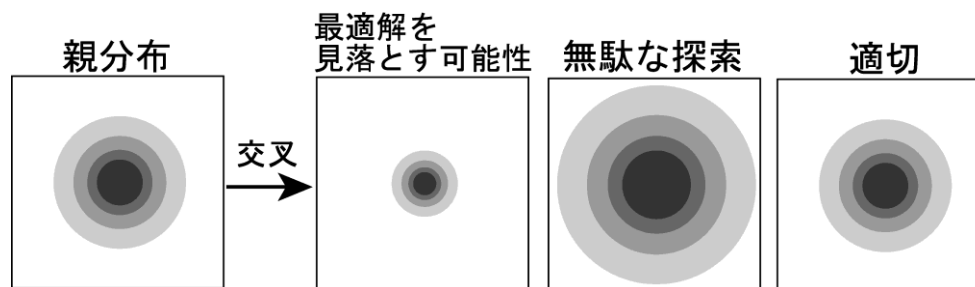
- 集団分布に従ってサンプリング領域を適応させる



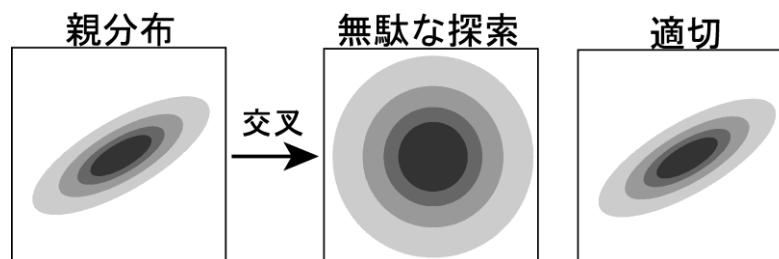
## 設計指針(2)

### ■ 交叉による効果的なサンプリング

変数間の  
依存関係が  
ない場合



変数間の  
依存関係が  
ある場合



### ■ 統計量の遺伝[喜多 99]

- 交叉により生成される子の分布は親の分布の平均, 分散を継承することが望ましい. 変数間に依存関係をもつ場合, 共分散も継承することが望ましい.



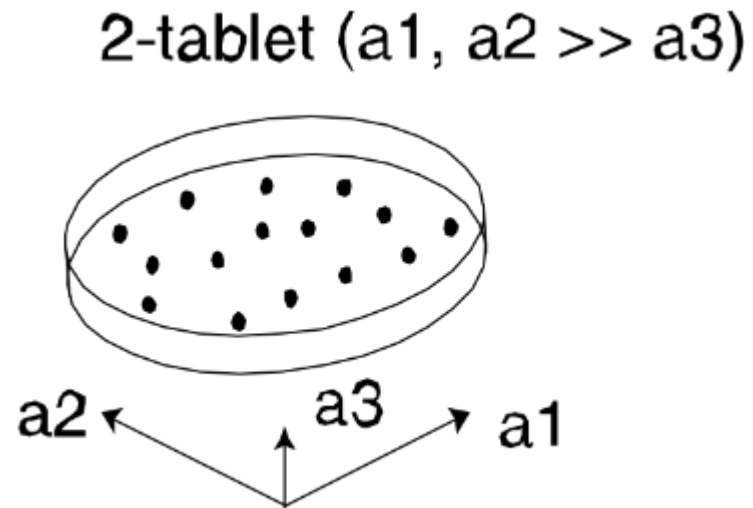
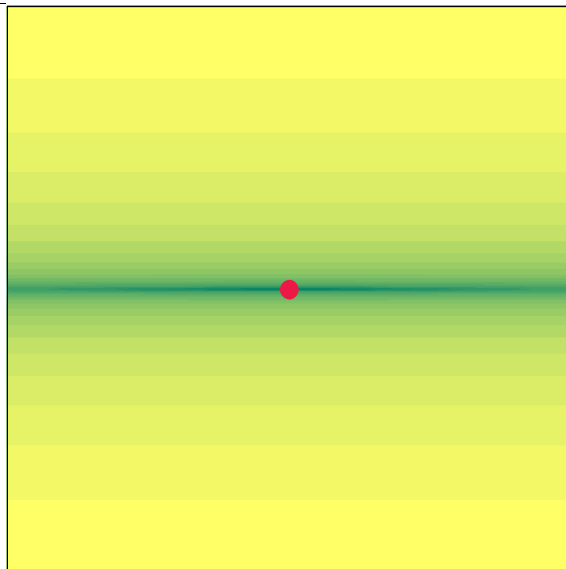


# 統計量の遺伝 [喜多 99] (1)

- 統計量の遺伝を満たす交叉は以下の式を満たす.
  - $E[X] = \mathbf{m} = \frac{1}{\mu} \sum_{j=1}^{\mu} \mathbf{y}_j$ ,
  - $Cov[X] = \frac{1}{\mu-1} \sum_{j=1}^{\mu} (\mathbf{y}_j - \mathbf{m})(\mathbf{y}_j - \mathbf{m})^T \cdots (\text{Eq.1})$
- ここで,
  - $X$  は親  $\mathbf{y}_1, \dots, \mathbf{y}_{\mu}$  と同じ分布に従う確率変数
  - $E[X]$  は  $X$  の期待値
  - $Cov[X] = E[(X - E[X])(X - E[X])^T]$  は共分散行列

# 交叉UNDXの問題点

- 悪スケール性を持つ関数において探索性能が劣化
  - 親1と親2を結ぶ軸に対して直交する部分空間 $S_{\perp}$ に等方的に子を生成
  - 部分空間 $S_{\perp}$ において悪スケール性→親に比べて極端に悪い子が生成





## REX [小林 09]

---

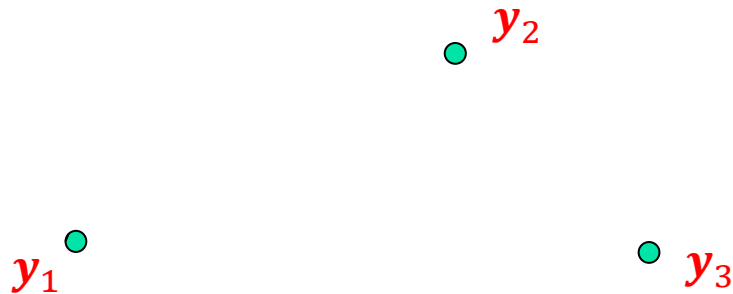
- REXは厳密に統計量を遺伝する.
  - $\mathbf{x}_i = \langle \mathbf{y} \rangle + \alpha \sum_{j=1}^{\mu} \epsilon_{i,j} (\mathbf{y}_j - \langle \mathbf{y} \rangle) \cdots$  (Eq.2)
    - $\mu$  : 親の数
    - $\mathbf{y}_1, \cdots, \mathbf{y}_{\mu}$  : 親
    - $\langle \mathbf{y} \rangle = \frac{1}{\mu} \sum_{j=1}^{\mu} \mathbf{y}_j$ .
    - $\alpha$  : 拡張率
    - $\epsilon$  : 平均0, 分散 $\sigma^2$ の独立同一分布に従う乱数



## REX [小林 09]

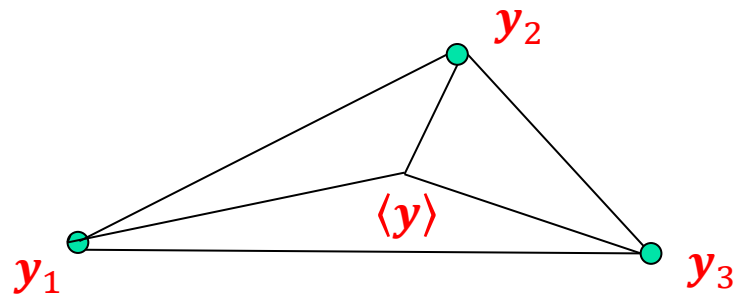
---

- REXは厳密に統計量を遺伝する.
  - $x_i = \langle y \rangle + \alpha \sum_{j=1}^{\mu} \epsilon_{i,j} (y_j - \langle y \rangle) \cdots$  (Eq.2)



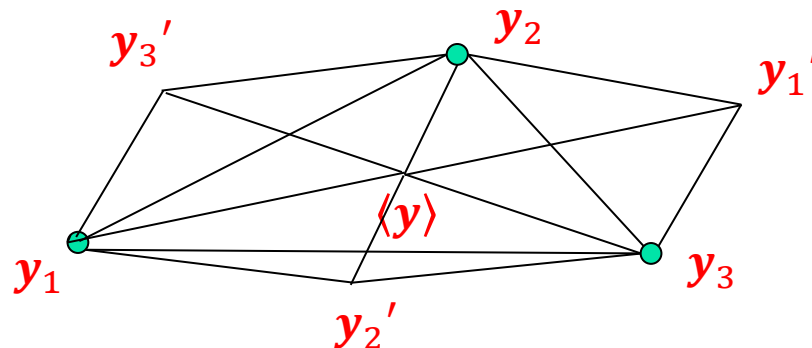
# REX [小林 09]

- REXは厳密に統計量を遺伝する.
  - $x_i = \langle y \rangle + \alpha \sum_{j=1}^{\mu} \epsilon_{i,j} (y_j - \langle y \rangle) \cdots$  (Eq.2)



# REX [小林 09]

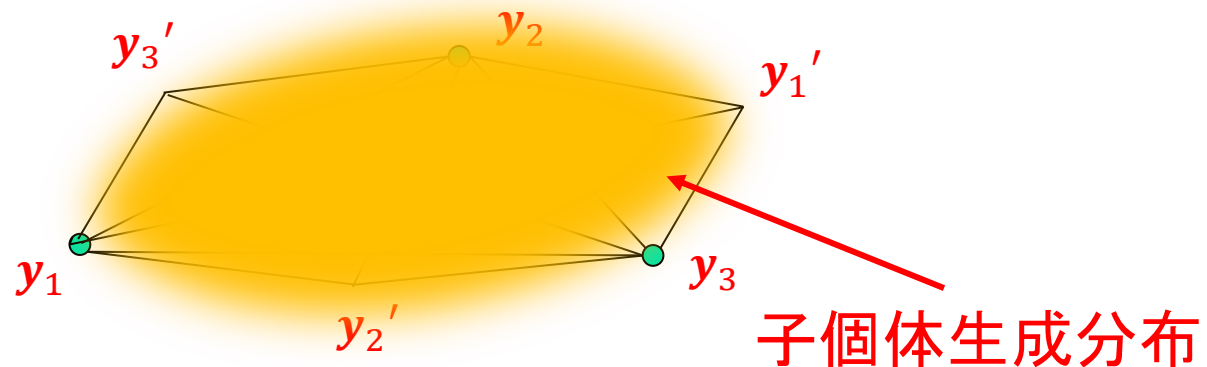
- REXは厳密に統計量を遺伝する.
  - $x_i = \langle y \rangle + \alpha \sum_{j=1}^{\mu} \epsilon_{i,j} (y_j - \langle y \rangle) \cdots$  (Eq.2)





# REX [小林 09]

- REXは厳密に統計量を遺伝する.
  - $x_i = \langle y \rangle + \alpha \sum_{j=1}^{\mu} \epsilon_{i,j} (y_j - \langle y \rangle) \cdots$  (Eq.2)





## REX [小林 09]

---

- $\alpha$  と  $\sigma$  の決定

- Eq.2に従って生成される子個体分布の共分散行列

- $Cov[X] = \alpha^2 \sigma^2 \sum_{j=1}^{\mu} (\mathbf{y}_i - \langle \mathbf{y} \rangle)(\mathbf{y}_i - \langle \mathbf{y} \rangle)^T \cdots \cdots (\text{Eq.3})$

- Eq.1 と Eq.3 を見比べると

- $\alpha = 1,$

- $\sigma = \sqrt{1/(\mu - 1)}.$



# JGG [Akimoto 07]

---

- アルゴリズム

1. 初期集団の生成

- $n_{\text{pop}}$  個の個体をランダムに生成する.

2. 複製選択

- $\mu$  個の親を集団からランダムに非復元抽出する.

3. 子の生成と評価

- 交叉により  $\lambda$  個の子を生成して評価する.

4. 生存選択

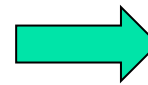
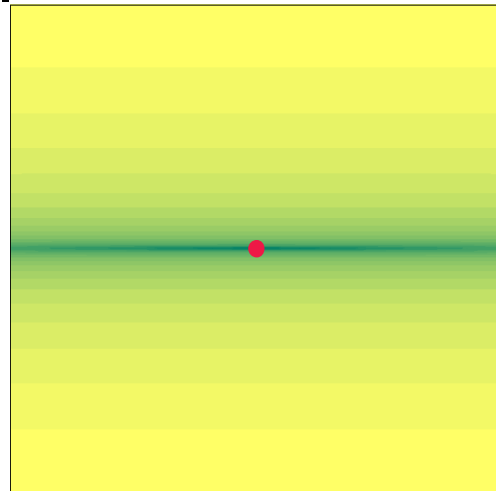
- 最良  $\mu$  個の子を集団中の  $\mu$  個の親と入れ替える.

5. 終了条件を満たすまで, ステップ2~4を繰り返す.

# REX/JGG と UNDX+MGG の性能比較

- 悪スケール性をもつベンチマーク関数

k-tablet

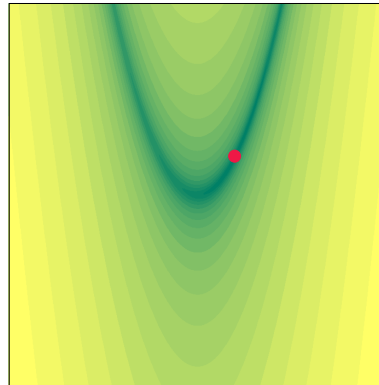


課題

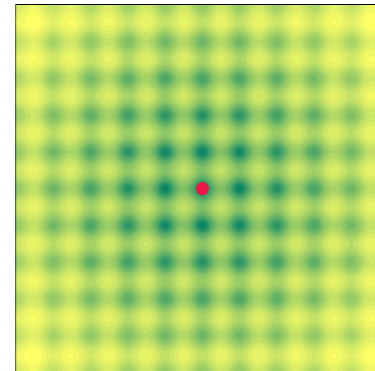
# REX/JGG と UNDX+MGG の性能比較

## ■ 他のベンチマーク関数

Rosenbrock



Rastrigin



関数名	$n$	REX/JGG	UNDX+MGG
Rosenbrock	20	$2.1 \times 10^5$	$1.2 \times 10^6$
Rastrigin	20	$3.1 \times 10^5$	$2.5 \times 10^6$

- $n$  : 次元
- REXの分布: 正規分布
- 親数:  $\mu = n + 1$



## 補足：Eclipseのインストール

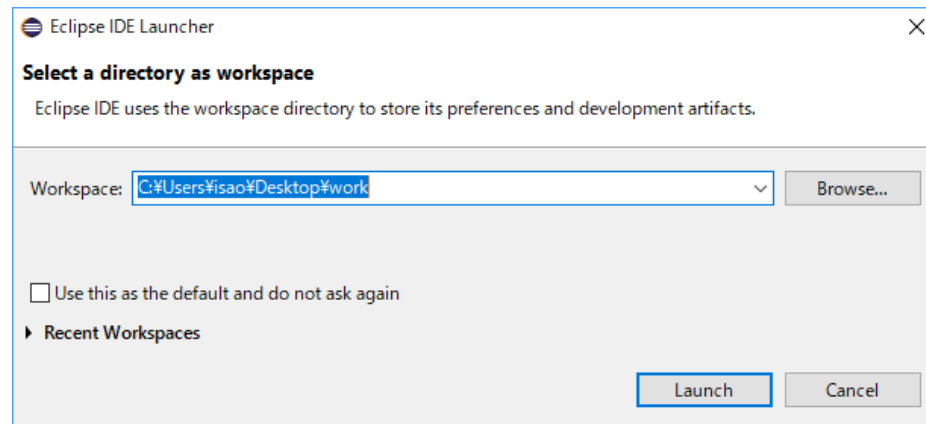
---

- 以下から最新版のEclipseをダウンロードしてインストールする
  - <https://www.eclipse.org/downloads/>



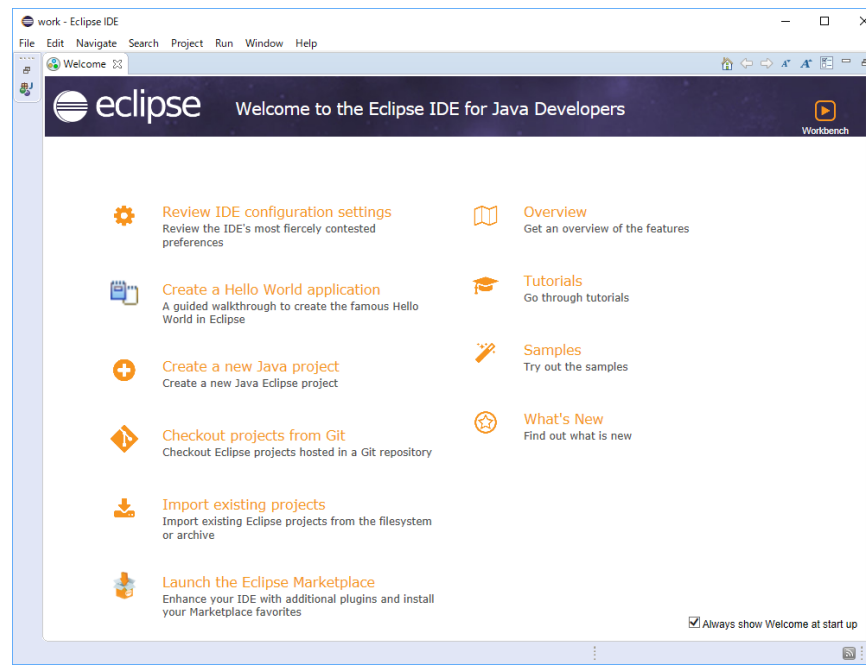
# 補足:プロジェクトのEclipseへのインポート(1)

- How to import the ResearchProject01 project into your Eclipse IDE
  - Put ResearchProject01.zip on your desktop.
  - Make a working folder on your desktop.
  - Start the Eclipse IDE.
  - You will get the "Eclipse IDE Launcher" dialog. Choose the working folder as "Workspace" and push the "Launch" button.



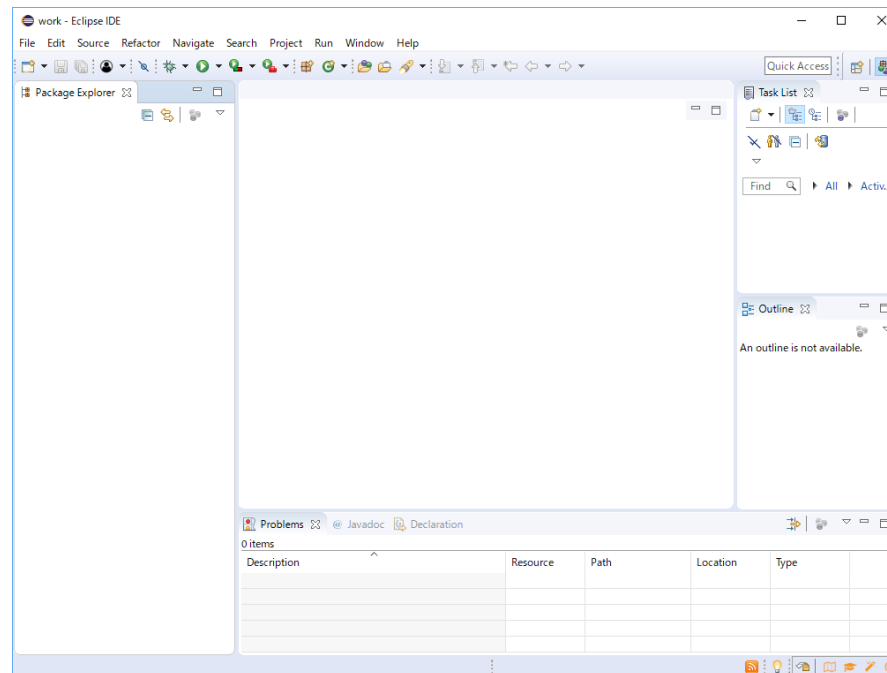
# 補足:プロジェクトのEclipseへのインポート(2)

- How to import the ResearchProject01 project into your Eclipse IDE
  - You will get the "Welcome" tab.
  - Click the "Workbench" button on the top right.



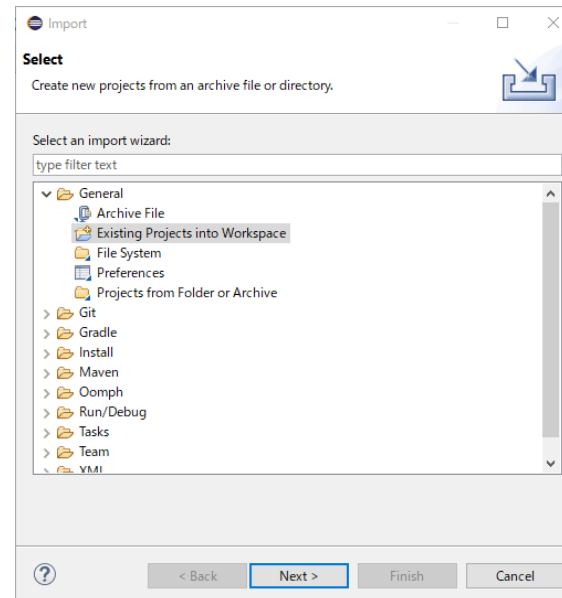
# 補足:プロジェクトのEclipseへのインポート(3)

- How to import the ResearchProject01 project into your Eclipse IDE
  - You will get the workbench.
  - Right click on the "Package Explorer" tab.



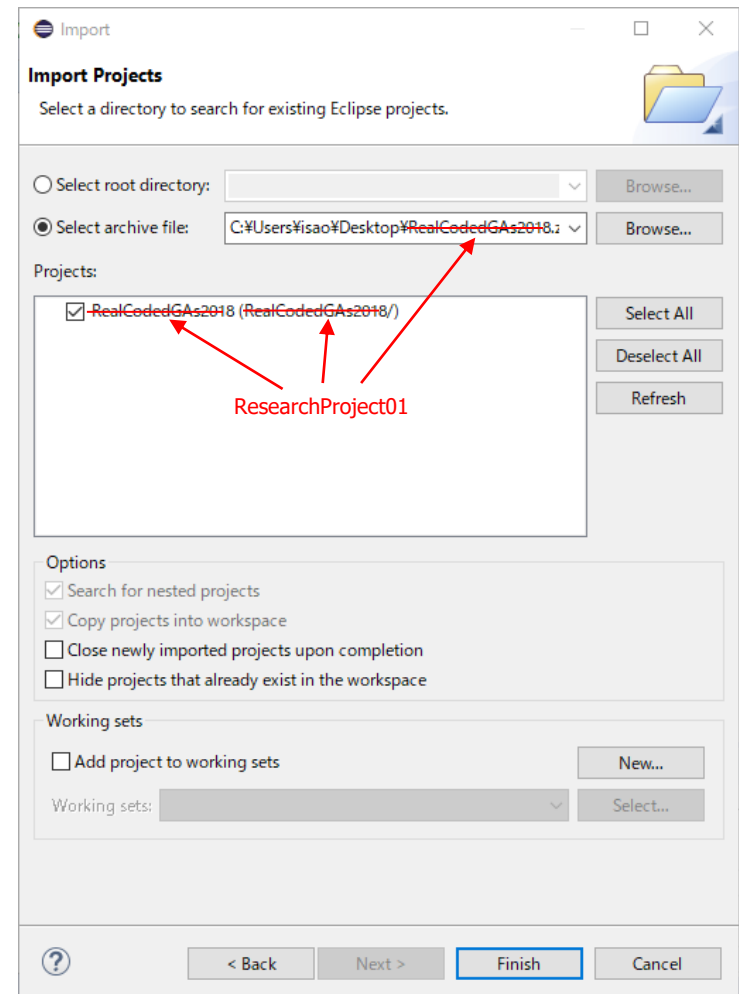
# 補足:プロジェクトのEclipseへのインポート(4)

- How to import the ResearchProject01 project into your Eclipse IDE
  - You will get a popup menu. Choose "import...".
  - You will get the "Import" dialog. Click "General" to open it and choose "Existing Projects into Workspace".
  - Click the "Next >" button.



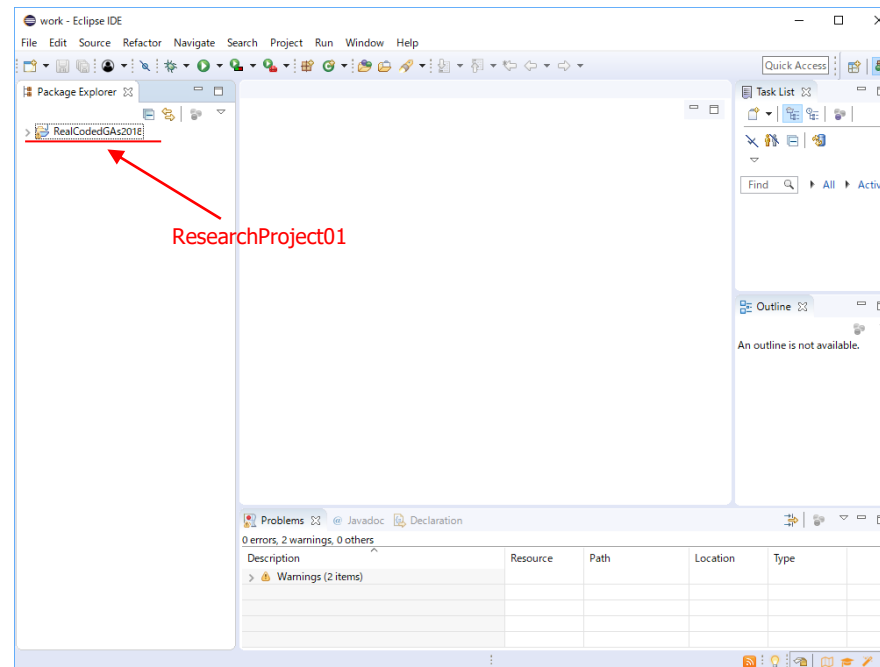
# 補足:プロジェクトのEclipseへのインポート(5)

- How to import the ResearchProject01 project into your Eclipse IDE
  - You will get the "Import Projects" dialog.
  - Click the "Select archive file:" and the "Browse...".
  - You will get "Select archive containing the projects to import" dialog. Choose the "ResearchProject01.zip" file on the desktop.
  - Click the "Finish" button.



# 補足:プロジェクトのEclipseへのインポート(6)

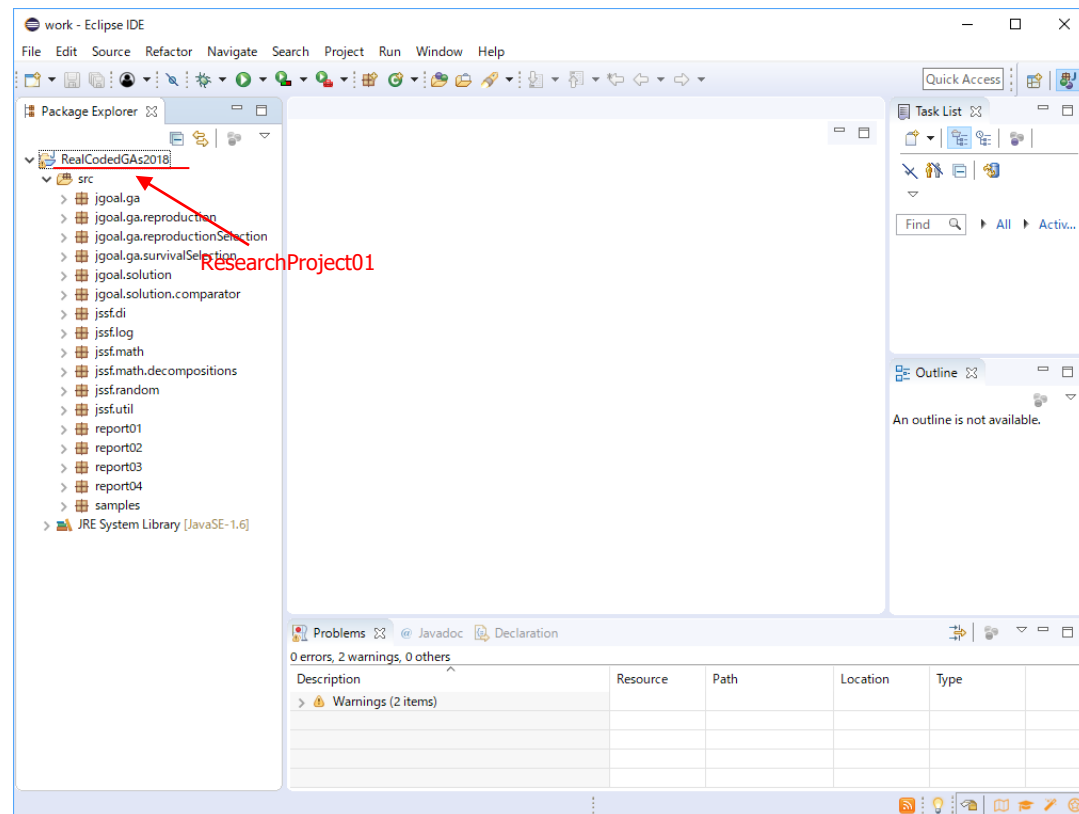
- How to import the ResearchProject01 project into your Eclipse IDE
  - You will find the " ResearchProject01" package on the "Package Explorer" tab.





# 補足:プロジェクトのEclipseへのインポート(7)

- Click the "ResearchProject01" icon on the package explorer and then click the "src" icon.





# 補足：プロジェクトのEclipseへのインポート(8)

---

- Packages for assignment
  - report01
    - Includes "TSRexNJggM.java" for assignment 1.
  - report02
    - Includes "TSRexNJggM.java" for assignment 2.
  - report03
    - Includes "TSRexNJggM.java" and "TSUndxMggM.java" for assignment 3.

# 補足：課題1のやり方(1)

- Assignment 1
  - Open the "TSRexNJggM.java" file in the "report01" package.

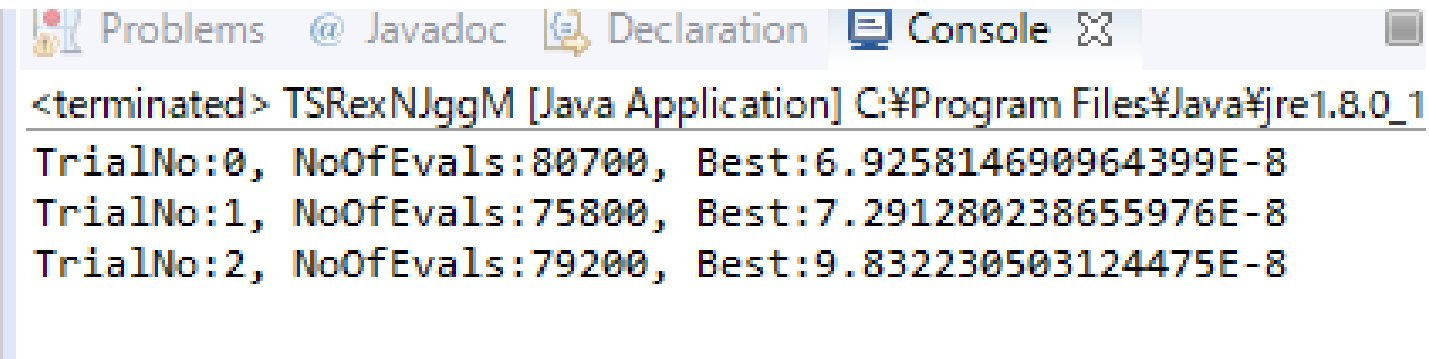
```
TSRexNJggM.java
1 package report01;
2
3 import java.io.IOException;
12
13 /**
14  * This program executes REX/JGG three trials, changing a random seed.
15  * It saves a transition of the best evaluation value at each generation in a log file.
16  * The log file is a CSV (comma separated values) file.
17  * The experimental settings is as follows.
18  * - Benchmark function: k-tablet ( $k=n/4$ ).
19  * - Dimension:  $n=20$ .
20  * - Initial region:  $[-5, +5]^n$ ,
21  * - Population size:  $14n$ 
22  * - The number of offspring:  $5n$ 
23  * - The maximum number of evaluation:  $4n \times 1e4$ 
24  * - The evaluation value for termination:  $1.0 \times 1e-7$ 
25  * - Log filename: RexJggKTabletP14K5.csv
26  *
27  * @author isao
28  *
29  */
30 public class TSRexNJggM {
31
```



# 補足：課題1のやり方(2)

## ■ Assignment 1

- Run "TSRexNJggM.java" by right-clicking on "TSRexNJggM.java" on the package explorer and choosing "Run As" -> "1 Java Application".
- You will get the following messages on the "Console" tab.

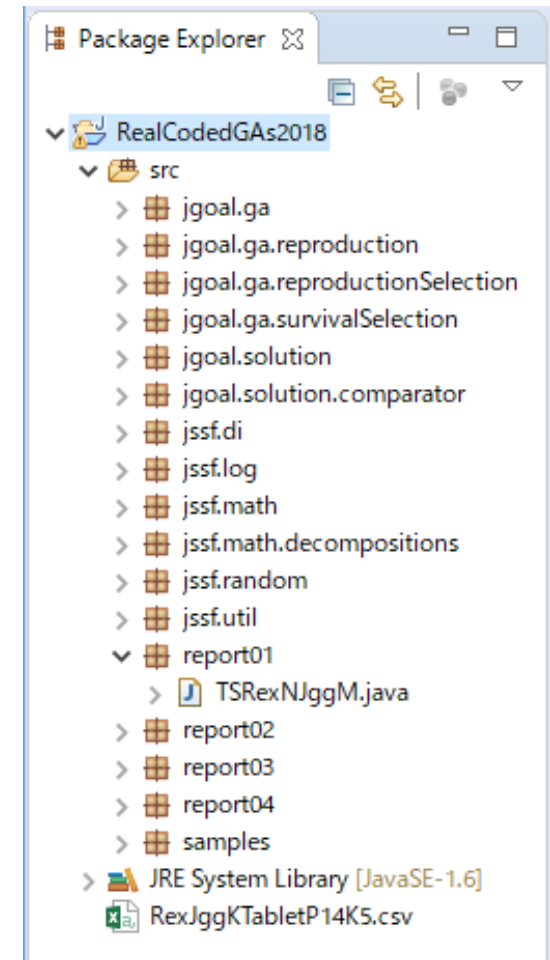


```
<terminated> TSRexNJggM [Java Application] C:\Program Files\Java\jre1.8.0_1
TrialNo:0, NoOfEvals:80700, Best:6.925814690964399E-8
TrialNo:1, NoOfEvals:75800, Best:7.291280238655976E-8
TrialNo:2, NoOfEvals:79200, Best:9.832230503124475E-8
```

# 補足：課題1のやり方(3)

## ■ Assignment 1

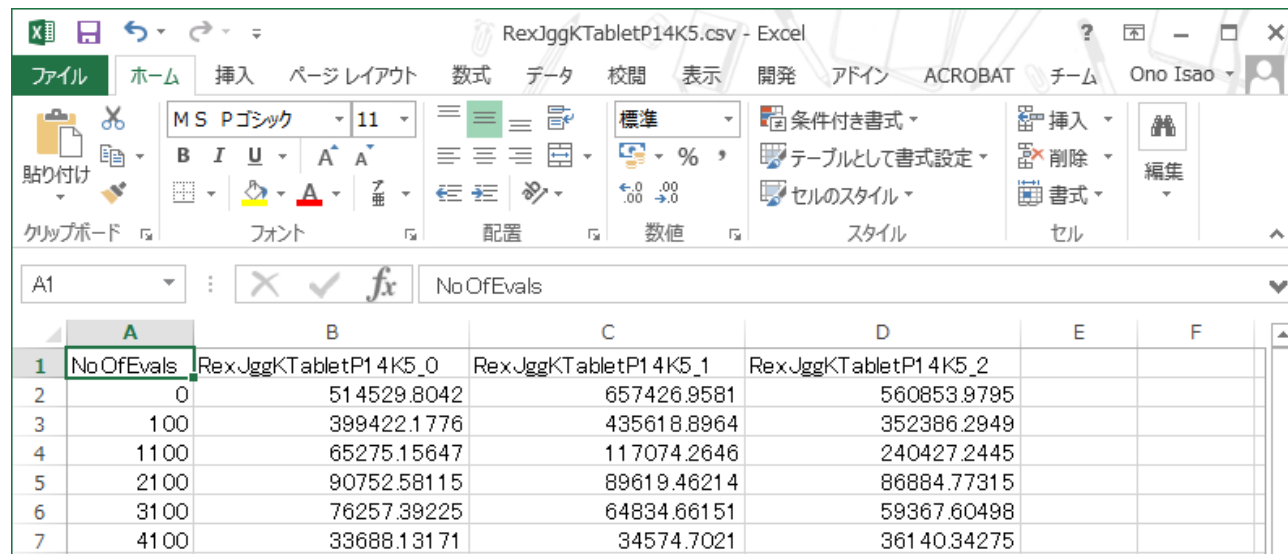
- The program generates a log file in the "comma separated values" (CSV) format which Excel can open.
- To show the log file, right-click on the "RealCodedGAs2018" icon on the package explorer and choose "Refresh", or push "F5".
- Then, you will find "RexJggKTabletP14K5.csv".



# 補足：課題1のやり方(4)

## ■ Assignment 1

- Opening "RexJggKTabletP14K5.csv" with Excel, you will find four columns.
  - The 1st column・・・the number of evaluation
  - The 2nd column・・・the evaluation values obtained in the 1st trial.
  - The 3rd column・・・the evaluation values obtained in the 2nd trial.
  - The 4th column・・・the evaluation values obtained in the 3rd trial.

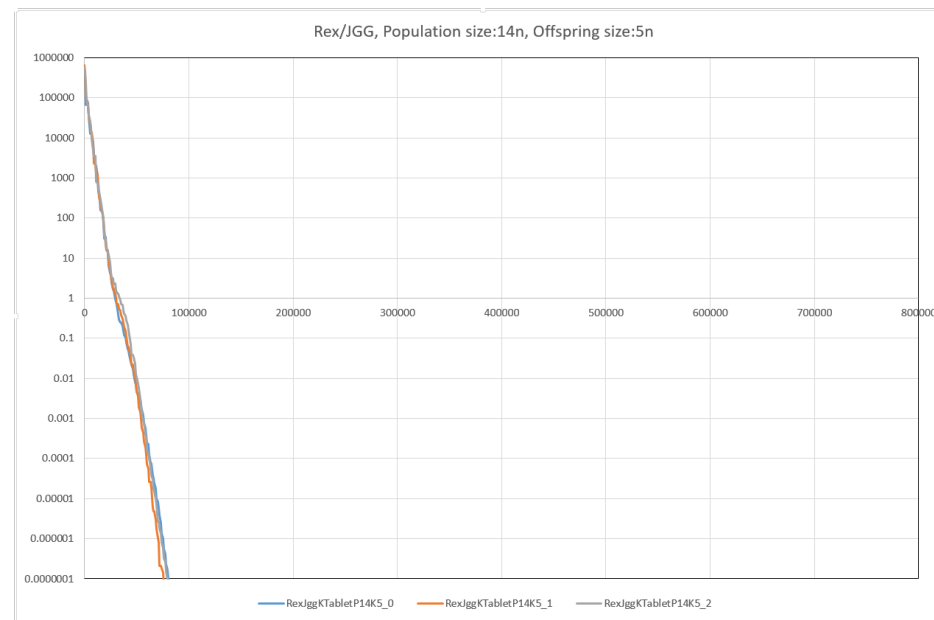


	A	B	C	D	E	F
1	NoOfEvals	RexJggKTabletP14K5_0	RexJggKTabletP14K5_1	RexJggKTabletP14K5_2		
2	0	514529.8042	657426.9581	560853.9795		
3	100	399422.1776	435618.8964	352386.2949		
4	1100	65275.15647	117074.2646	240427.2445		
5	2100	90752.58115	89619.46214	86884.77315		
6	3100	76257.39225	64834.66151	59367.60498		
7	4100	33688.13171	34574.7021	36140.34275		

# 補足：課題1のやり方(5)

## ■ Assignment 1

- Using Excel or some tools like gnuplot, you can plot the convergence curves.
  - Set the vertical axis to the logarithmic scale.
  - Use the same scales when comparing more than two graphs.







# 補足：課題1のやり方(6)

- Assignment 1
  - Change the following lines:
    - Line 127: 14 -> 7, 28
    - Line 133: RexJggKTabletP14K5 -> RexJggKTabletP7K5, RexJggKTabletP28K5

```
124 public static void main(String[] args) throws IOException {  
125     boolean minimization = true; //Minimizes the objective function.  
126     int dimension = 20; //Dimension  
127     int populationSize = 14 * dimension; //The population size  
128     int noOfKids = 5 * dimension; //The number of offspring  
129     double min = -5.00; //The minimum value of the initialization area.  
130     double max = +5.00; //The maximum value of the initialization area.  
131     long maxEvals = (long)(4 * dimension * 1e4); //The maximum number of evaluation  
132     int maxTrials = 3; //The number of trials  
133     String trialName = "RexJggKTabletP14K5"; //The trial name, which is used for the log filename.  
134     String logFilename = trialName + ".csv"; //The log filename.
```



## 補足：課題4のやり方

- Make a new package and copy the "TSRexNJggM.java" and "TSUndxMggM.java" to the new package, and change the "ktablet" method in each file.

```
50
57- /**
58     * k-tablet function (k=n/4)
59     * @param x n-dimensional real-valued vector
60     */
61- private static double ktablet(TCMatrix x) {
62     int k = (int)((double)x.getDimension() / 4.0); //k=n/4
63     double result = 0.0; //Initializes the result evaluation value as zero.
64     for (int i = 0; i < x.getDimension(); ++i) {
65         double xi = x.getValue(i); //i-th element of x.
66         if (i < k) {
67             result += xi * xi;
68         } else {
69             result += 10000.0 * xi * xi;
70         }
71     }
72     return result;
73 }
```