OpenMDAO を用いた最適化とその評価

2014/11/14

オープン CAE シンポジウム 2014

坪田遼

概要

1. OpenMDAO の紹介
2. 最適化とは
3. 最適化の分類
4. 最適化実施手順
5. ベンチマーク
    1. ベンチマーク例題
    2. 実験計画法ごとの違い
    3. メタモデルごとの違い
    4. 最適化手法ごとの違い

OpenMDAO の紹介

- 複合領域設計解析 / 最適化フレームワーク
  (**M**ultidisciplinary **D**esign **A**nalysis and **O**ptimization)
- 開発元は NASA グレンリサーチセンター
- ライセンスは Apach Licence 2.0
- Windows / Linux / OS X 対応
- http://openmdao.org/

OpenMDAO の紹介 - 特徴

- 複数のソフトウェアを組み合わせて 1 つのシステムを作成可能（連成解析、最適化・・・）
- Python ベースのフレームワーク
- 最適化用エンジン
- Chrome ベースの GUI
- コマンドラインでの実行も可能

操作例については以下を参照
「 OpenMDAO 入門 」
（ 2014/3/8 第 28 回 オープン CAE 勉強会 @ 関西）
http://www.slideshare.net/HTsubota/open-mdao1

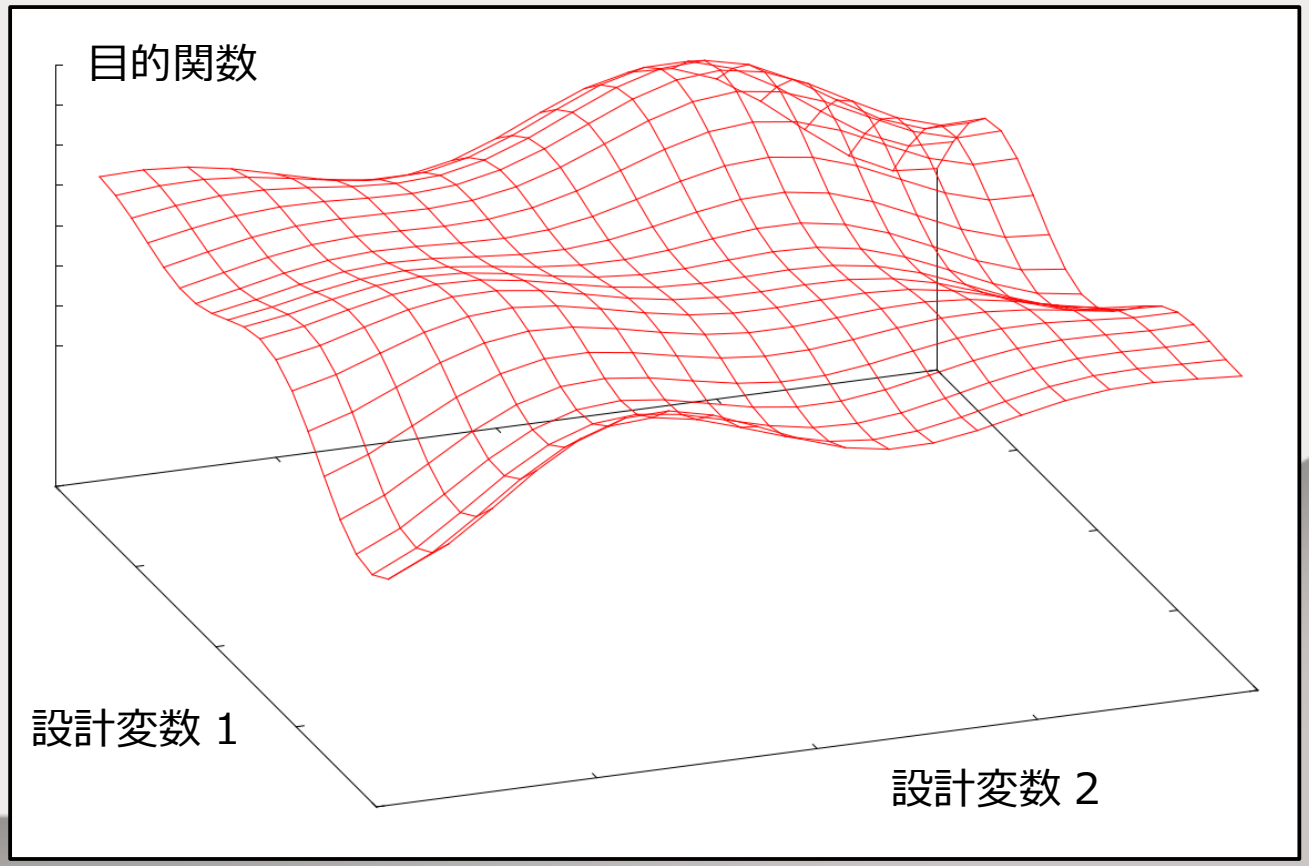
最適化とは

設計変数（入力）
- 梁の太さ
- 管の径、曲げ角度
- 発熱体の位置 …

システム
（構造物、配管、熱伝導体、・・・）

目的関数（出力）
- 応力、重量
- 圧損、流速
- 最大温度 …

①
設計変数に対する
目的関数の形状を推定
②
最も適切な設計変数を
見つけ出す



最適化の分類

	局所的	大域的
単目的	・目的関数が１つ ・設計パラメーターの初期値から勾配をたどる	・目的関数が１つ ・一定範囲内の目的関数形状を推定する
多目的	・目的関数が複数 ・設計パラメーターの初期値から勾配をたどる	・目的関数が複数 ・一定範囲内の目的関数形状を推定する

局所・・単峰性問題（単純な目的関数形状）に使用。計算量は少ない。
大域・・多峰性問題（複雑な目的関数形状）に使用。計算量は多い。
単目的・・最適値が一意に存在
多目的・・トレードオフが発生する場合がある（パレート最適）。

最適化の分類

	局所的	大域的
単目的	・目的関数が１つ ・設計パラメーターの初期値から勾配をたどる	・目的関数が１つ ・一定範囲内の目的関数形状を推定する
多目的	・目的関数が複数 ・設計パラメーターの初期値から勾配をたどる	・目的関数が複数 ・一定範囲内の目的関数形状を推定する

局所・・単峰性問題（単純な目的関数形状）に使用。計算量は少ない。
大域・・多峰性問題（複雑な目的関数形状）に使用。計算量は多い。
単目的・・最適解が一意に存在
多目的・・トレードオフが発生する場合がある（パレート最適）。

最適化の実施手順

1. 実験計画法
実際に実験（計算）するサンプリング点を設定

2. メタモデルの作成
サンプリング点での実験値（計算値）から
目的関数形状を推定

3. 最適化
メタモデルから条件を満たす（最適な）解を
得られる設計変数を推定

4. 結果の確認
得られた設計変数で実験（計算）をして
問題ないことを確認

※1

OpenMDAO 標準機能

- 等分割
- 一様乱数
- ラテン超方格
- 最適ラテン超方格
（ Optimal Latin Hypercube ）
- 中心複合計画
- CSV ファイル指定

適切なサンプリング点を選択し、
無駄な解析を避けます。

最適化の実施手順

1. 実験計画法
実際に実験（計算）するサンプリング点を設定

2. メタモデルの作成
サンプリング点での実験値（計算値）から
目的関数形状を推定

3. 最適化
メタモデルから条件を満たす（最適な）解を
得られる設計変数を推定

4. 結果の確認
得られた設計変数で実験（計算）をして
問題ないことを確認

※2

OpenMDAO 標準機能

・クリギング
・ロジスティック回帰
・ 2 次応答曲面

メタモデルによって実際の
解析回数を減らします。

最適化の実施手順

1. 実験計画法
実際に実験（計算）するサンプリング点を設定

2. メタモデルの作成
サンプリング点での実験値（計算値）から
目的関数形状を推定

3. 最適化
メタモデルから条件を満たす（最適な）解を
得られる設計変数を推定

4. 結果の確認
得られた設計変数で実験（計算）をして
問題ないことを確認

※3

OpenMDAO 標準機能

- COBYLA*1
- CONMIN*2
- 遺伝的アルゴリズム
- NEWSUMT*3
- SLSQP*4

また公式プラグインで pyOpt の ALPSO 、 SNOPT 、 NSGA2 なども使用できます。

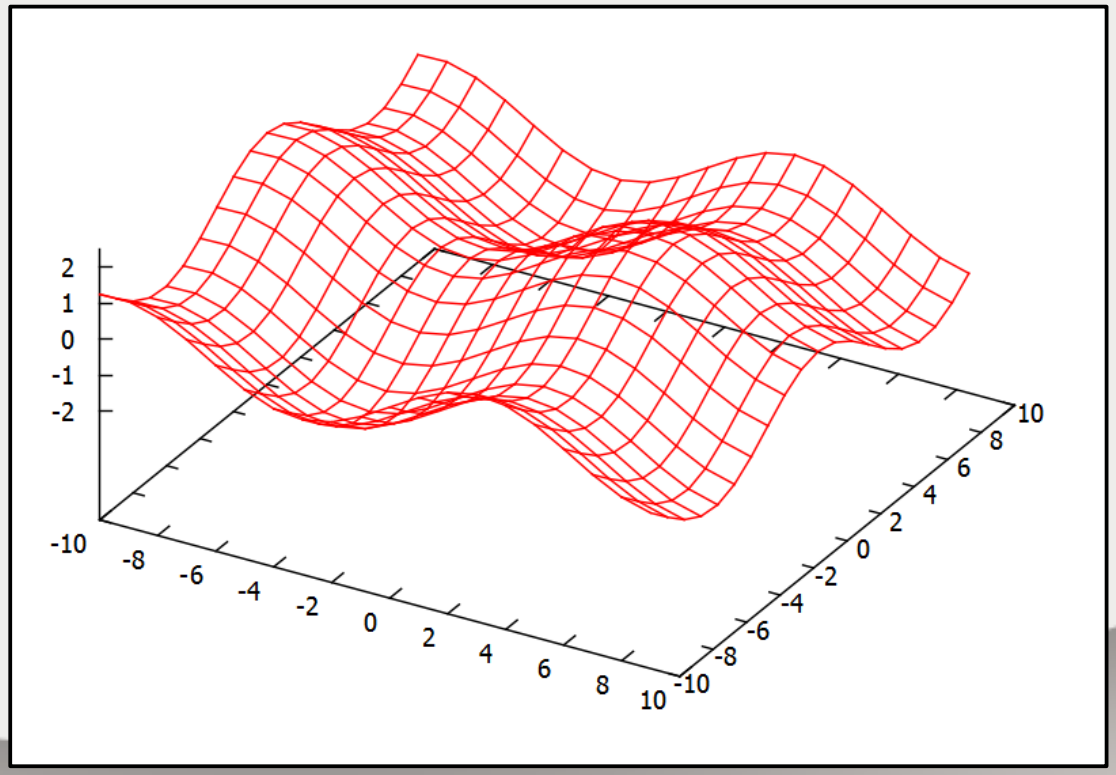
*1 線形近似による制約付き最適化
*2 制約付き関数最小化
*3 ニュートン法：制約なし逐次最小化
*4 逐次最小二次計画法

ベンチマーク例題

$f(x, y) = \sin(0.5x) + \cos(0.5y)$

$-10 < x < 10$

$-10 < y < 10$

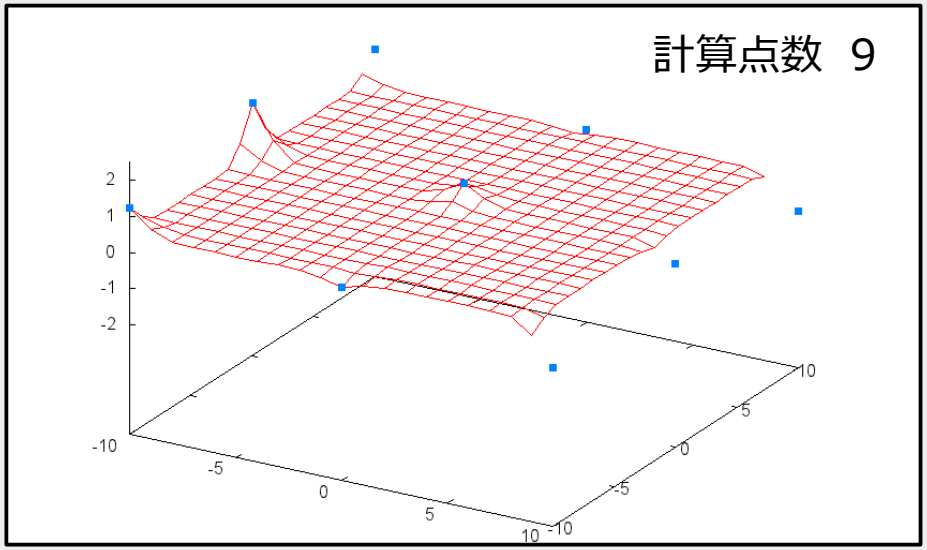


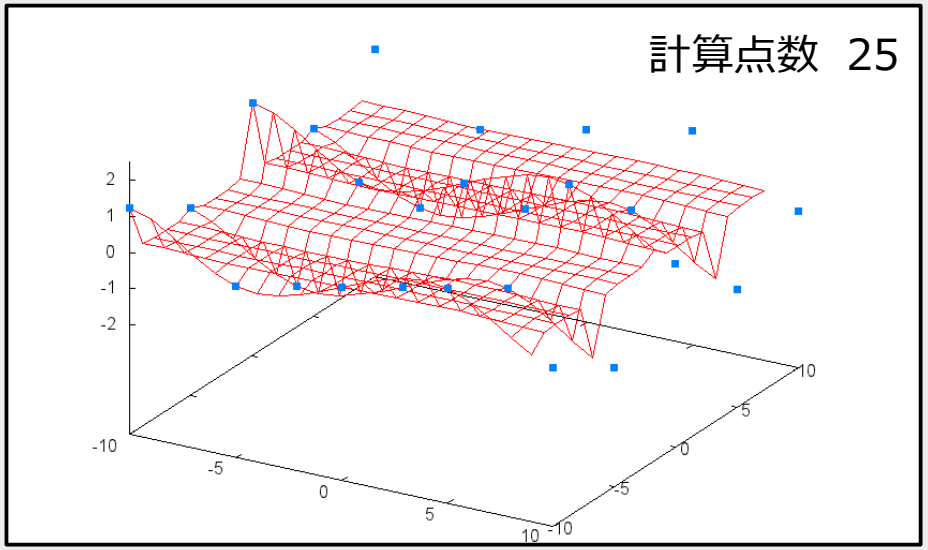
実際の目的関数形状
（計算点 400 ）

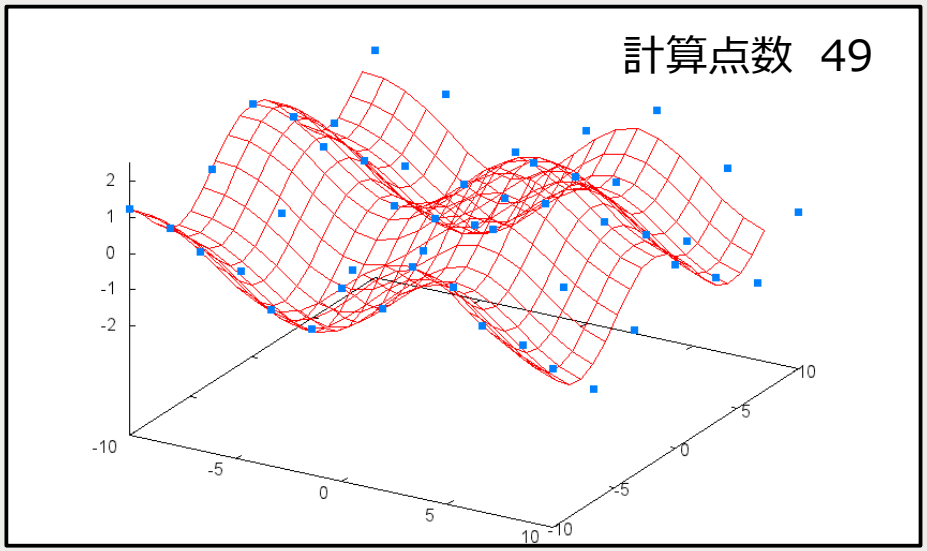
実験計画法ごとの違い

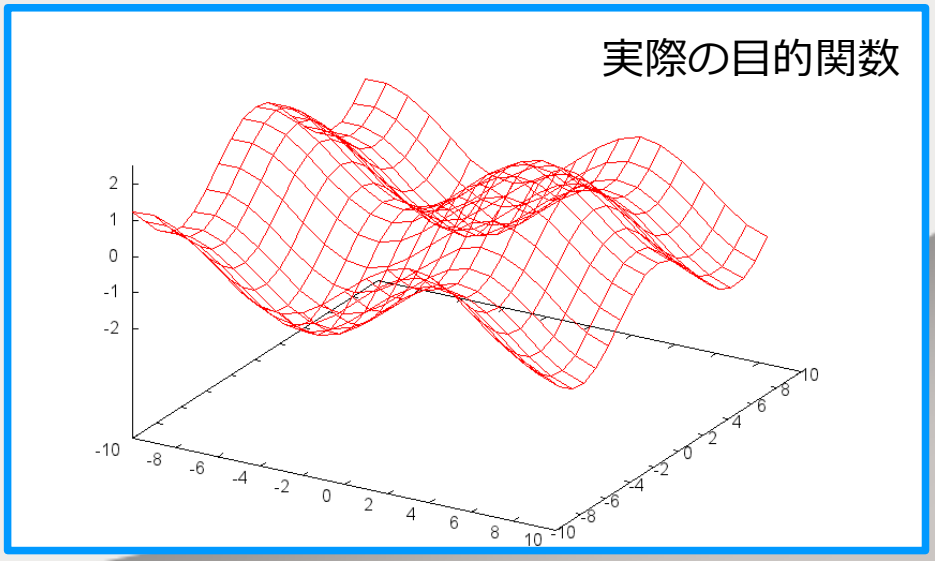
・クリギング法で目的関数形状を推定

等分割





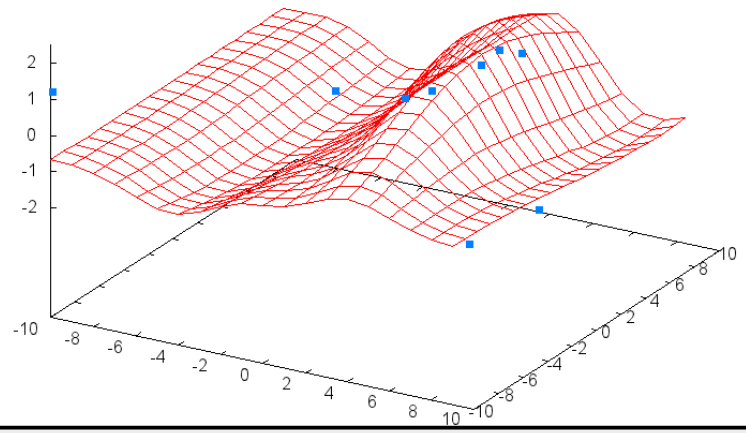




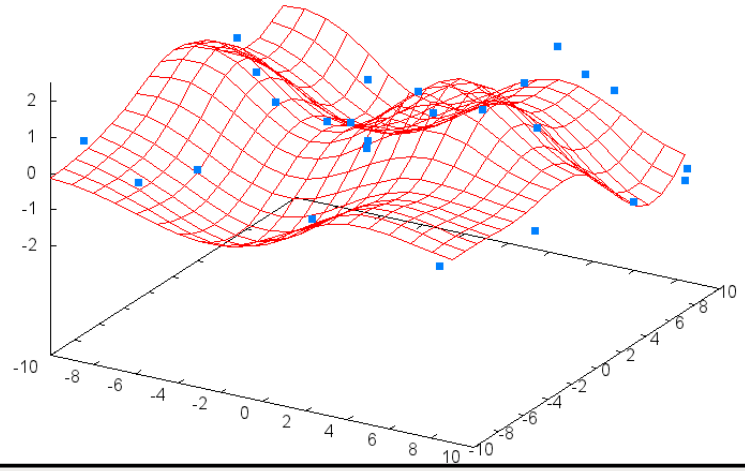
※ メタモデル作成方法はクリギング法

一様乱数

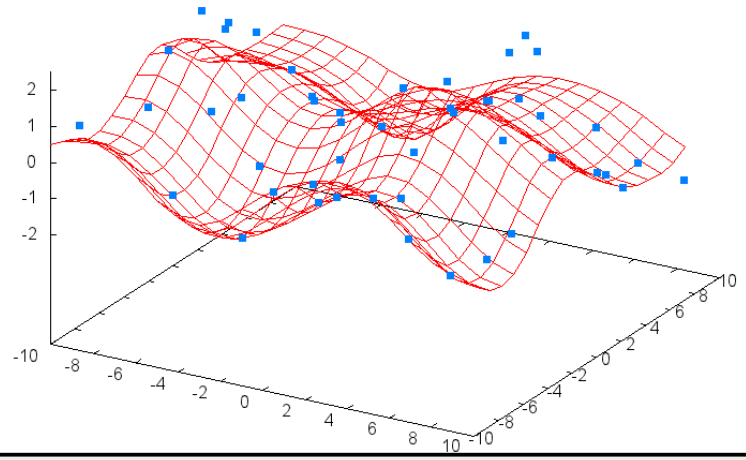
計算点数　9



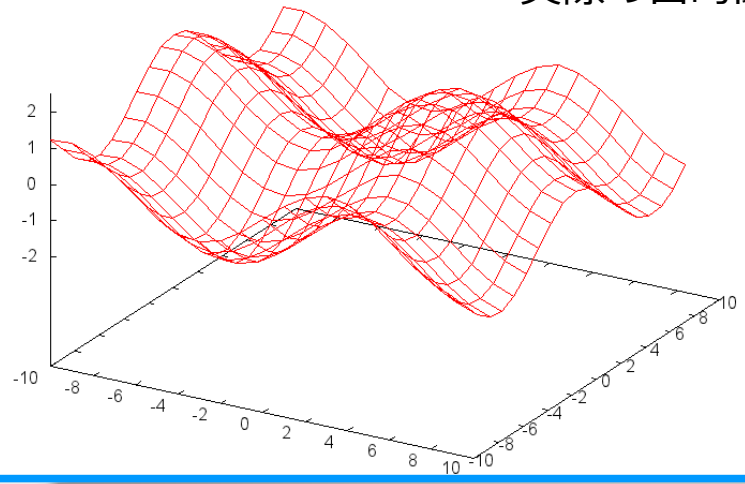
計算点数　25



計算点数　49



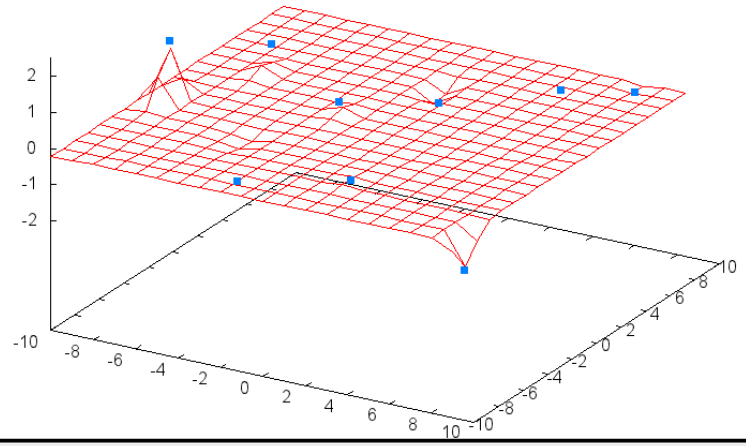
実際の目的関数



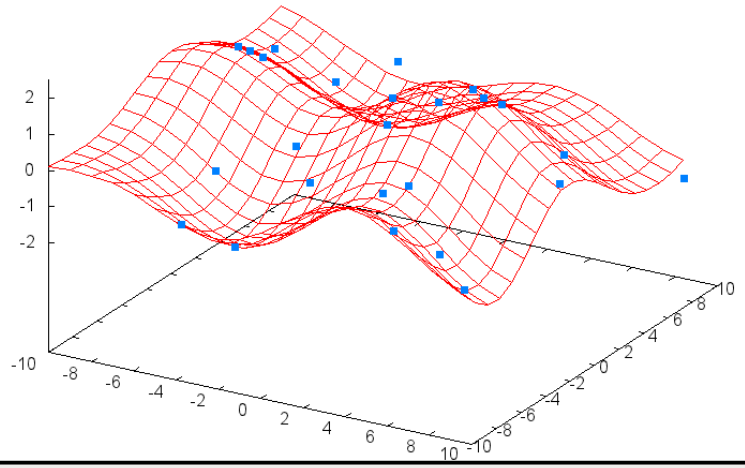
※ メタモデル作成方法はクリギング法

ラテン超方格

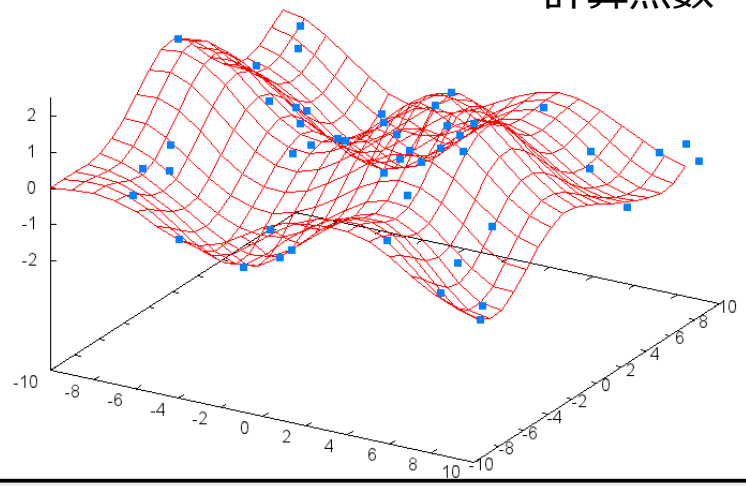
計算点数　9



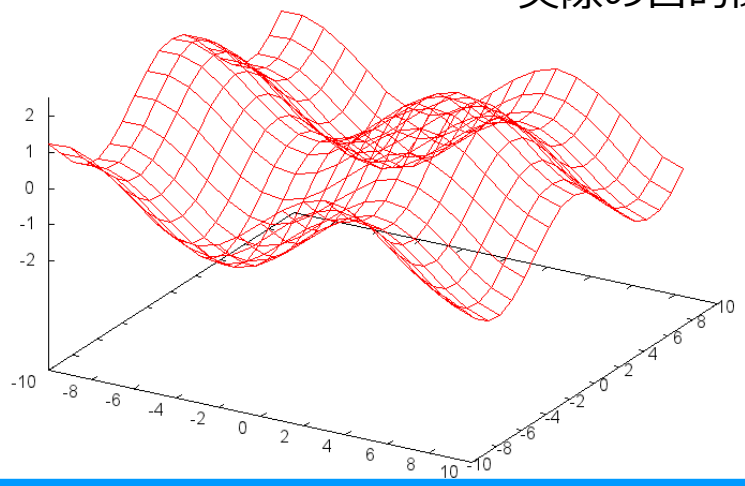
計算点数　25



計算点数　49



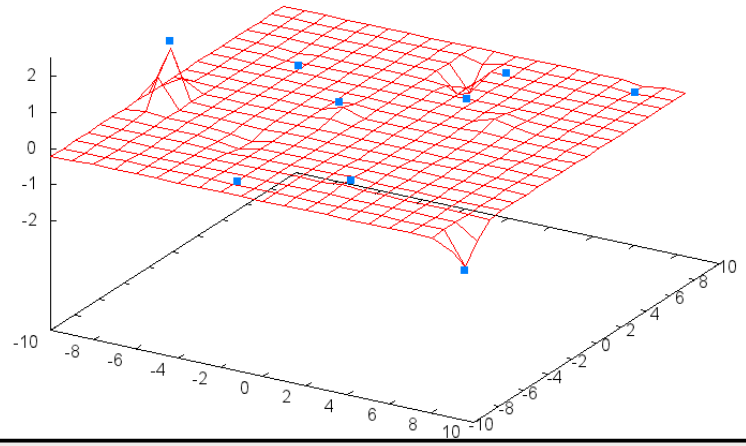
実際の目的関数



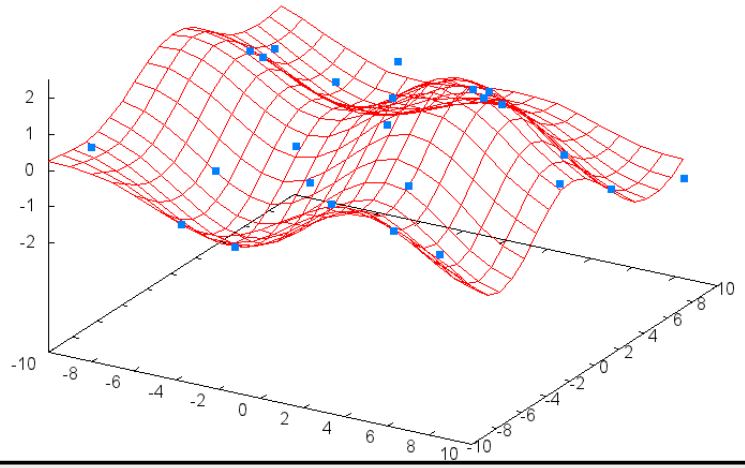
※ メタモデル作成方法はクリギング法

最適ラテン超方格

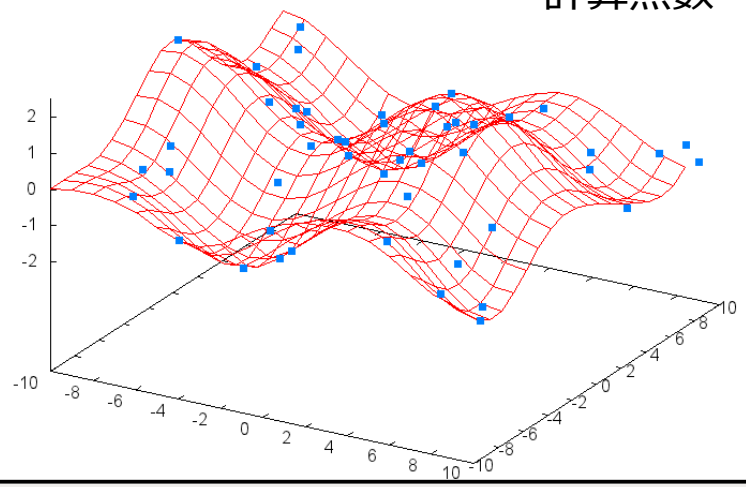
計算点数　9



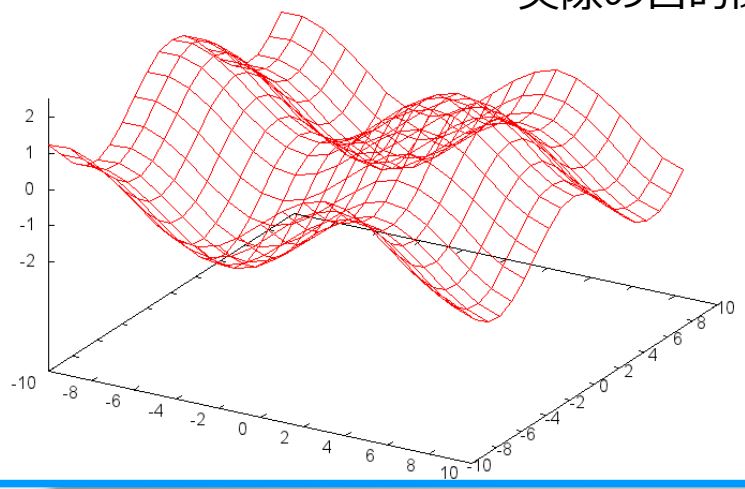
計算点数　25



計算点数　49

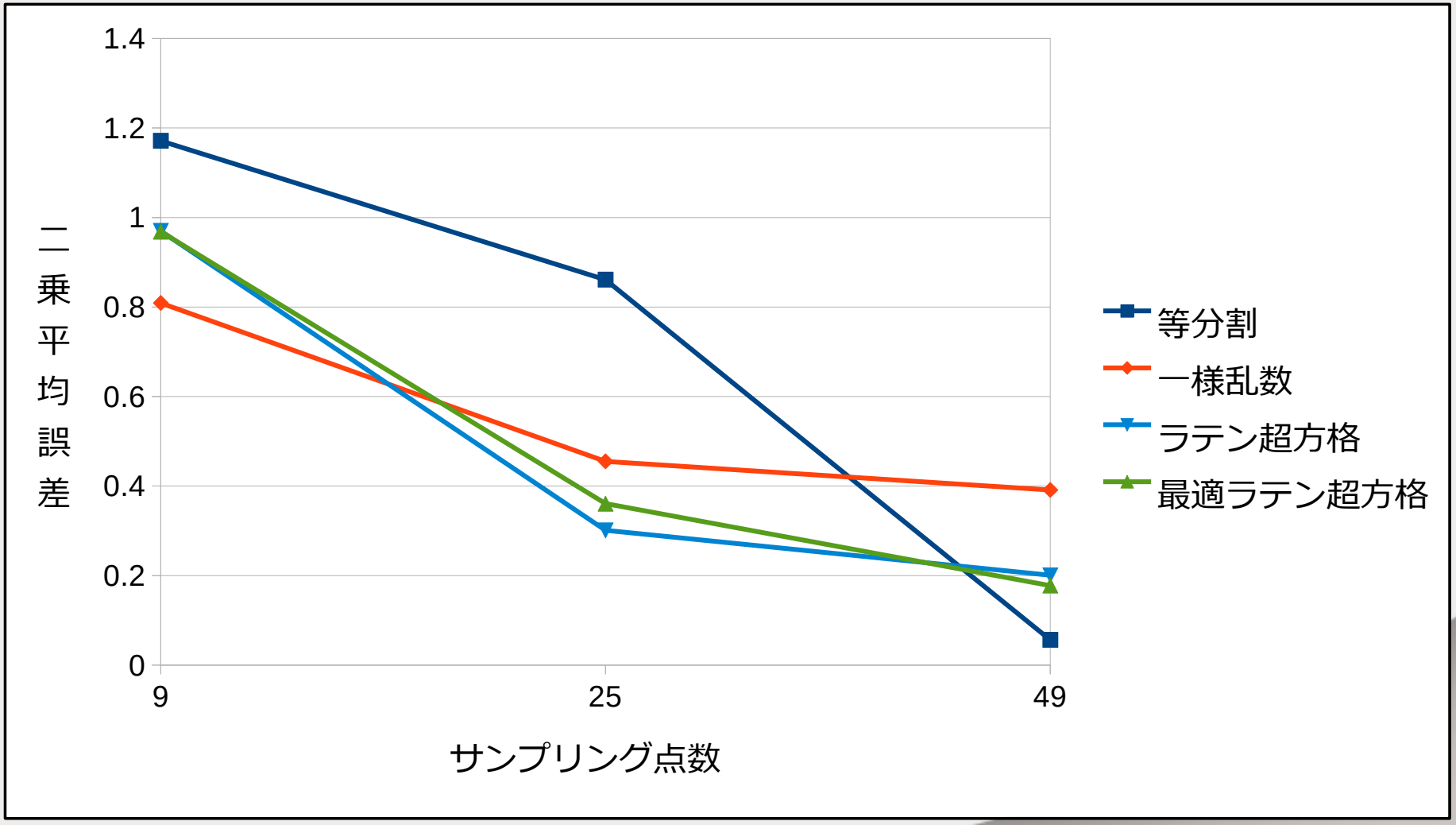


実際の目的関数



※ メタモデル作成方法はクリギング法

実験計画法ごとのメタモデルの誤差

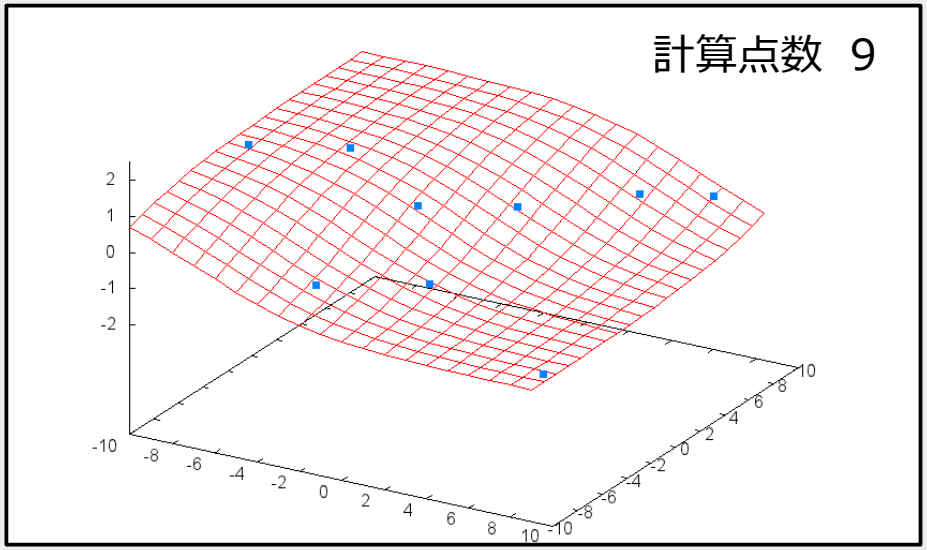


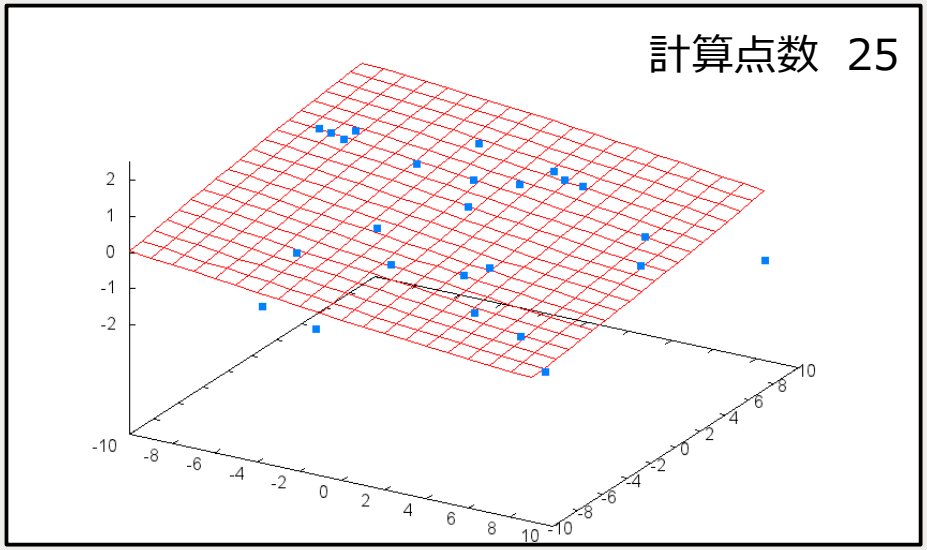
※ メタモデル作成方法はクリギング法

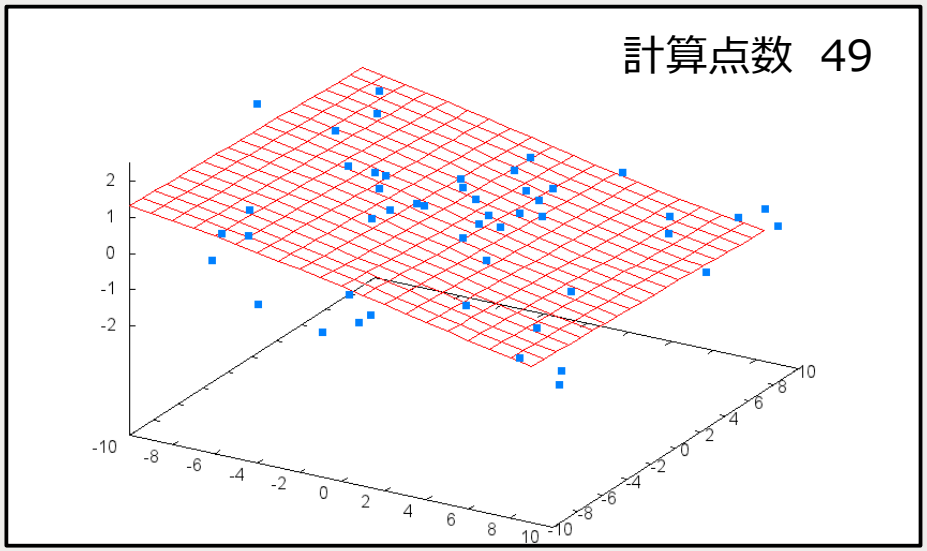
メタモデルごとの違い

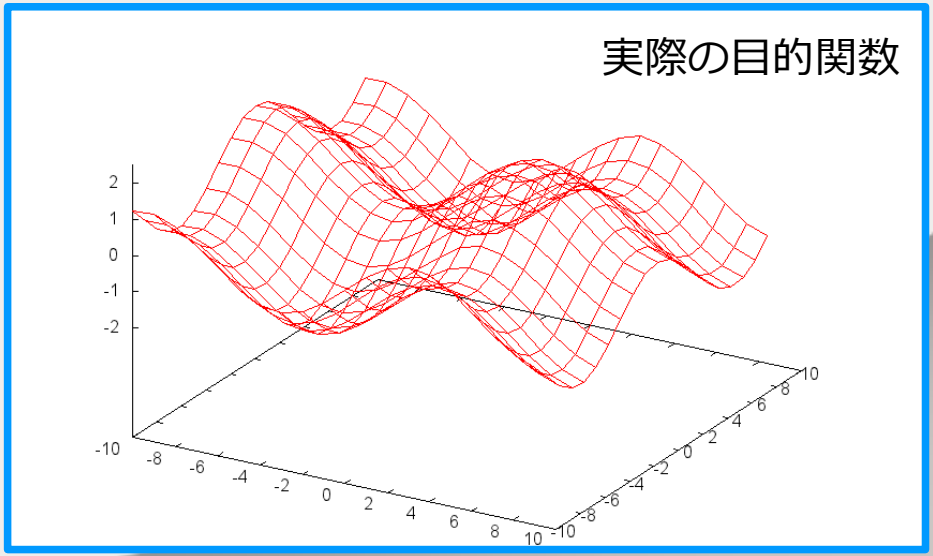
・ラテン超方格によるサンプリング点から計算

ロジスティック回帰





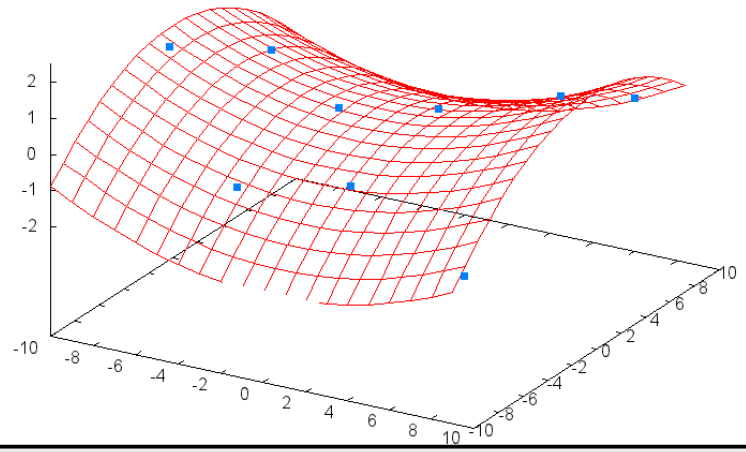




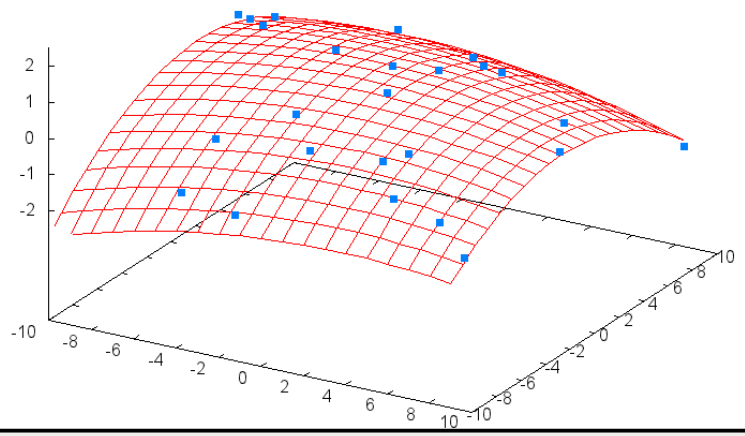
※ 実験計画法はラテン超方格

2 次応答曲面

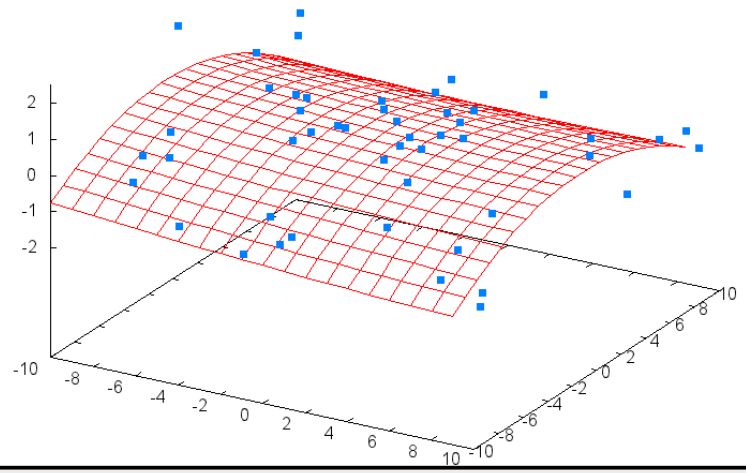
計算点数　9



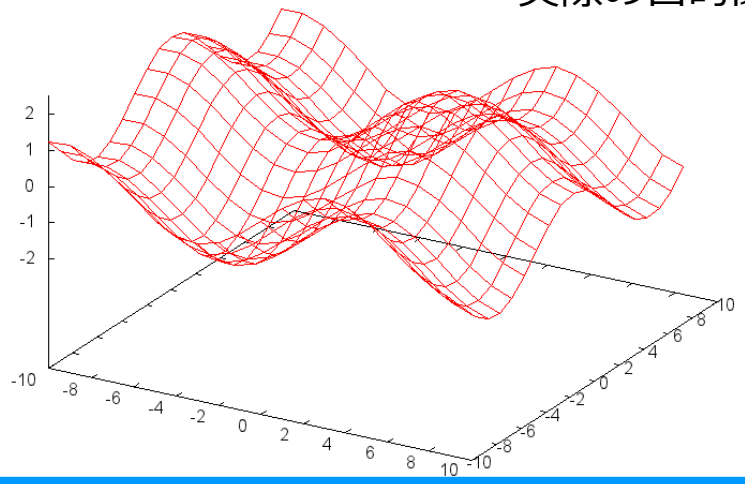
計算点数　25



計算点数　49



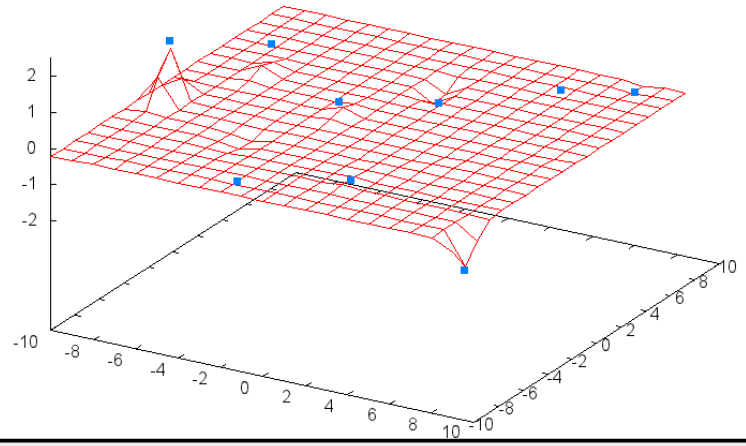
実際の目的関数



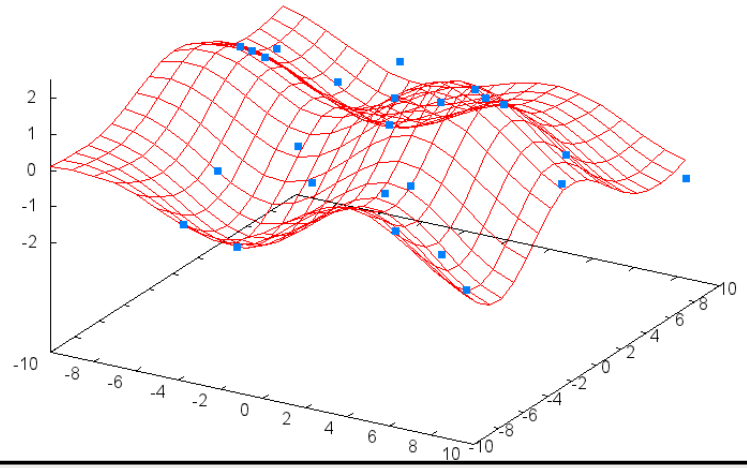
※ 実験計画法はラテン超方格

クリギング

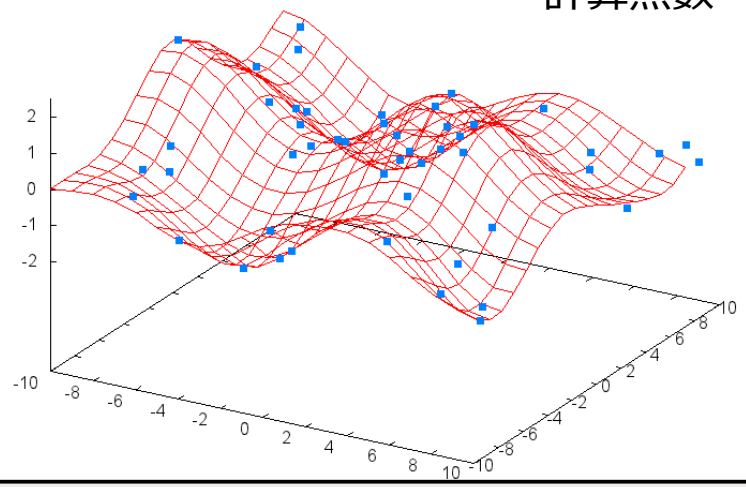
計算点数　9



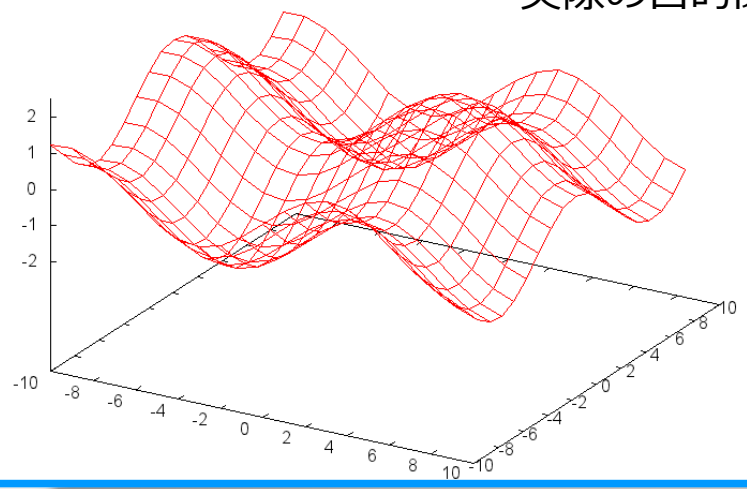
計算点数　25



計算点数　49



実際の目的関数



※ 実験計画法はラテン超方格

最適化手法ごとの違い

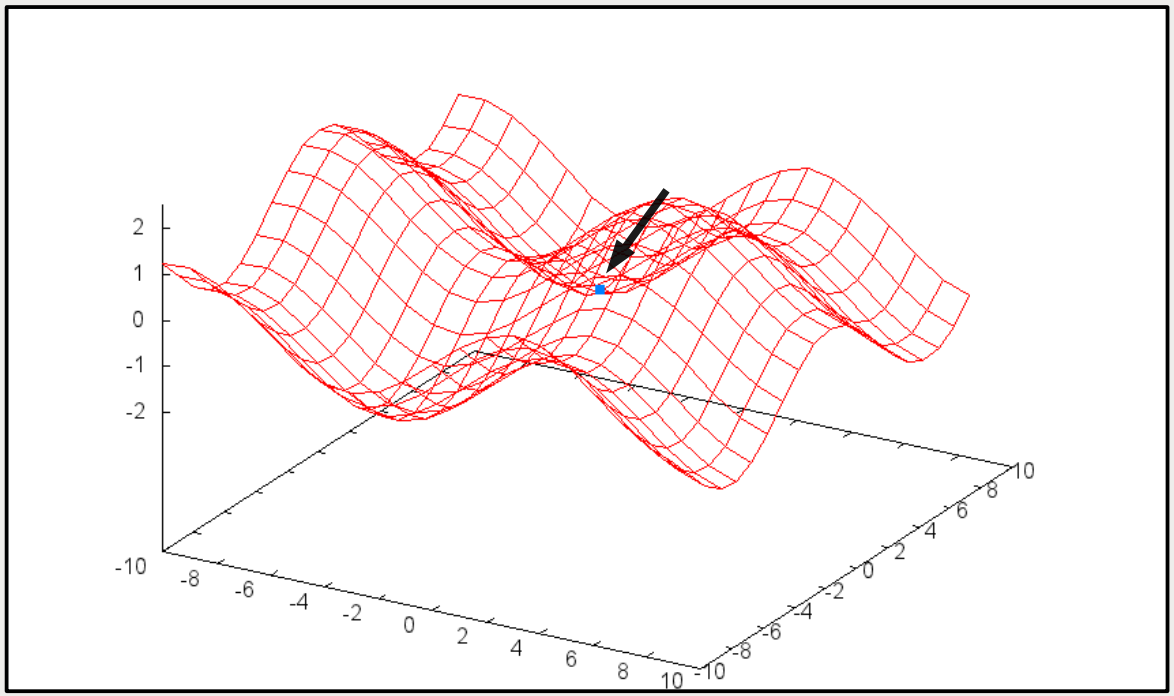
・最小値を探索

・実モデルで実行（メタモデル不使用）

・パラメーターはデフォルト

COBYLA

Constrained **O**ptimization **BY** **L**inear **A**pproximation



実験回数　50 回

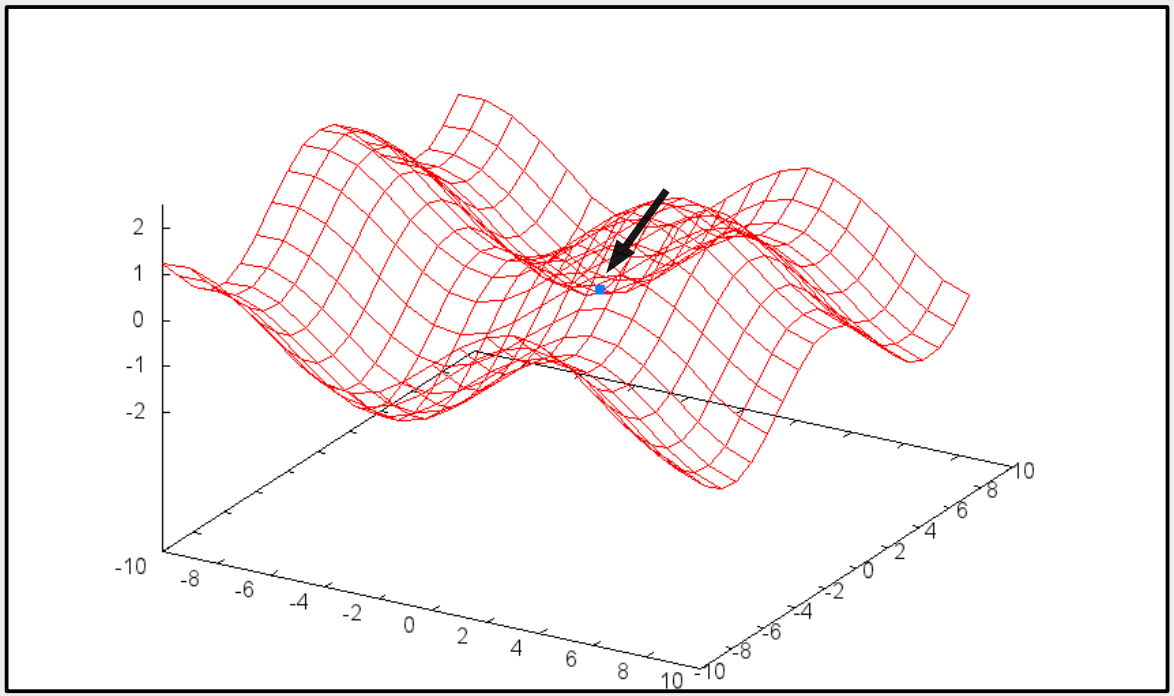
最適値　-2.0

設計パラメーター

（ -3.141677, 6.283240 ）

CONMIN

CONstrained function **MIN**imization (Feasible direction method)



実験回数　65 回

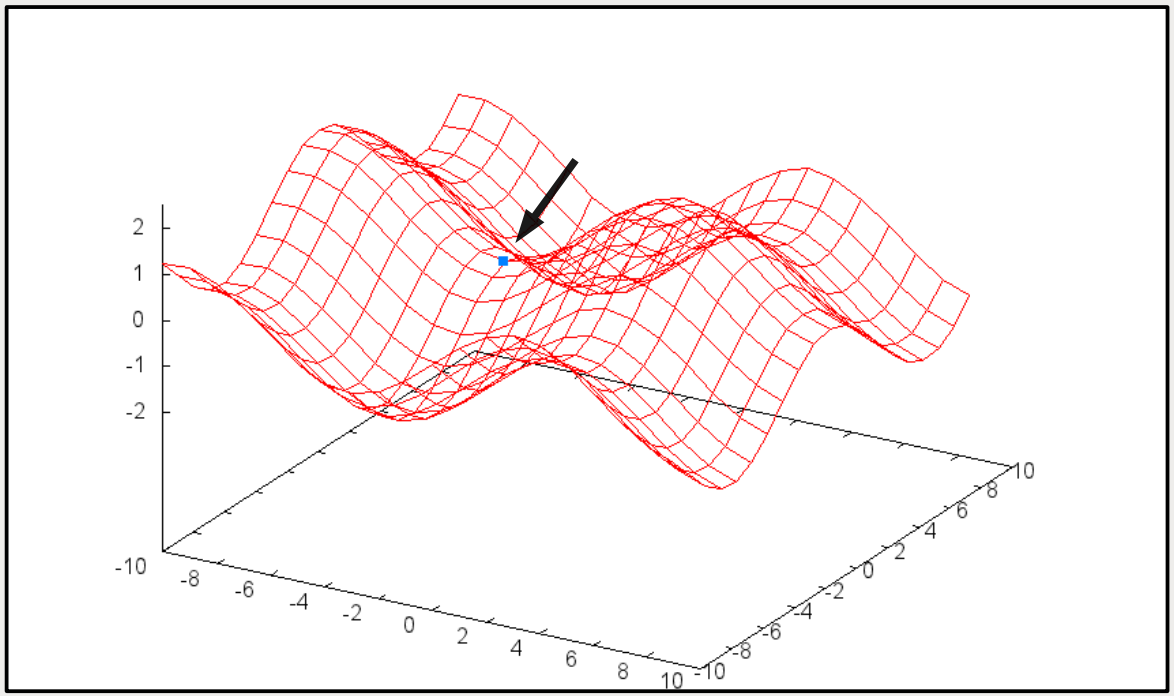
最適値　-2.0

設計パラメーター

（ -3.141593, 6.283185 ）

NEWSUMT

NEWton's method **S**equence of **U**nconstrained **M**inimizations



実験回数　189 回

最適値　0.0

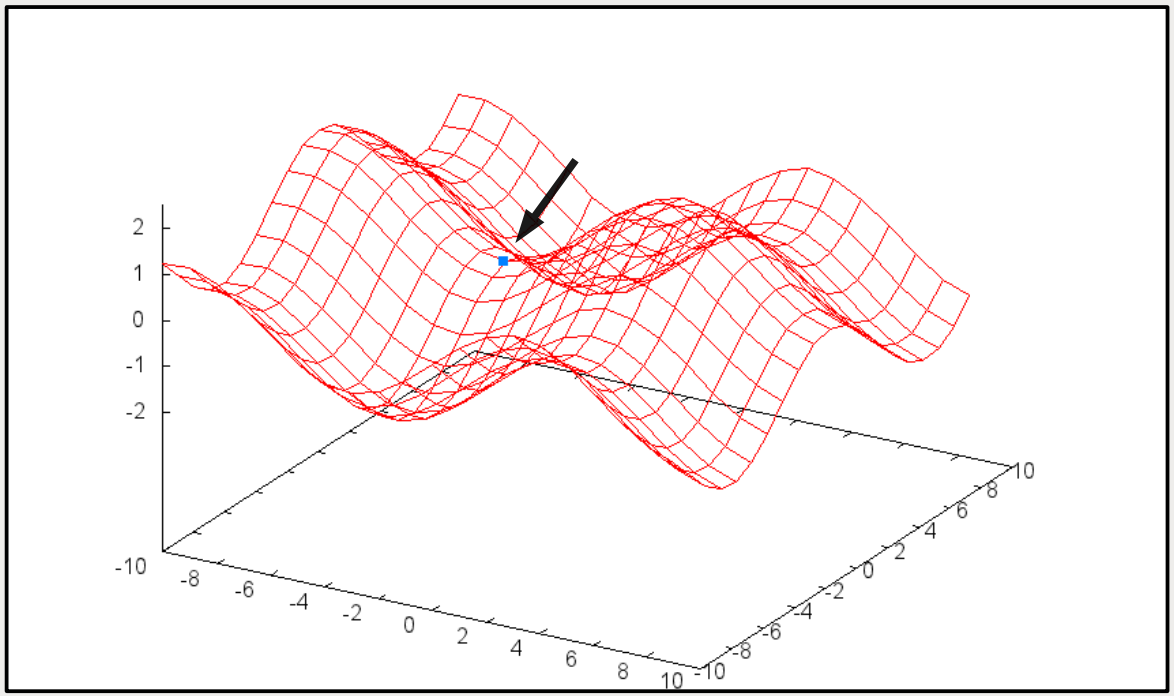
設計パラメーター

（ -3.141372, 0.0 ）

不適切な解

SLSQP

Sequential **L**east **SQ**uares **P**rogramming



実験回数　16 回

最適値　0.0

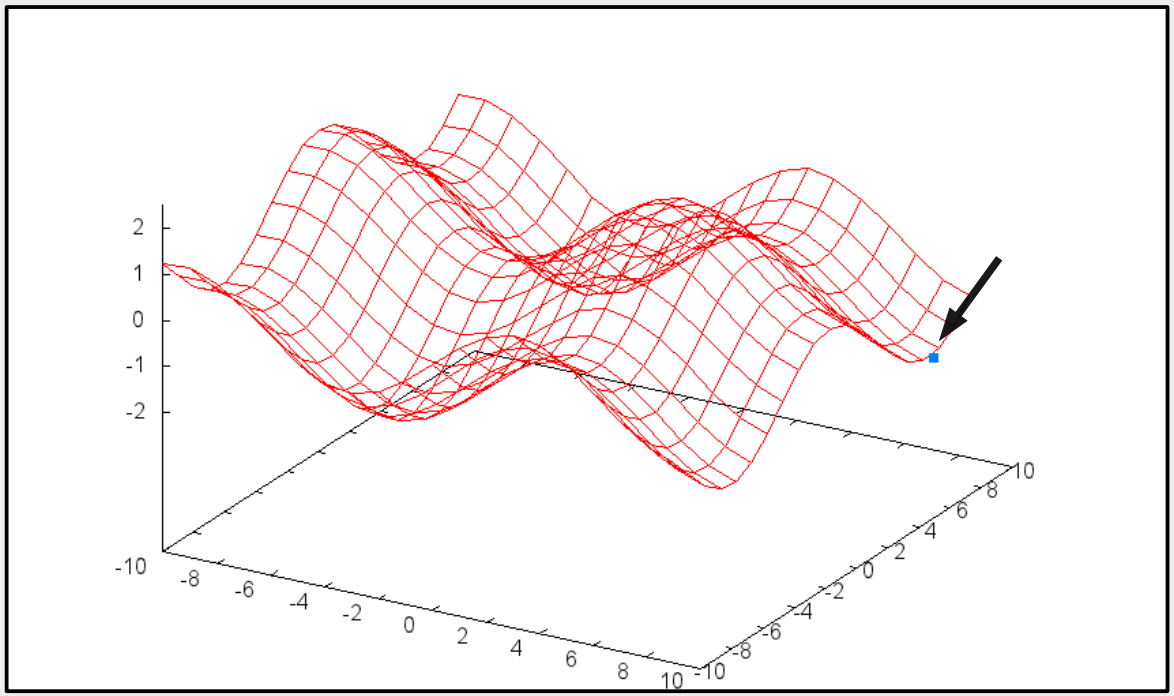
設計パラメーター

（ -3.141487, 0.000002 ）

不適切な解

遺伝的アルゴリズム

General genetic algorithm



実験回数　8081 回

最適値　-1.989295

設計パラメーター

（ 9.162881, 6.414148 ）

まとめ

- 実験計画法
  ラテン超方格系が比較的安定
  計算量は 1/5 程度まで減らせる
- メタモデル
  目的関数形が不明な場合はクリギング法
- 最適化手法
  COBYLA または CONMIN

  計算量を許容できれば遺伝的アルゴリズム

参照

※1）実験計画法 - OpenMDAO で使用できる標準機能

- Building a Model - Executing a Design of Experiment (DOE)
  http://openmdao.org/docs/tutorials/optimization/doe.html
- Package openmdao.lib - DOEGENERATORS
  http://openmdao.org/docs/srcdocs/packages/openmdao.lib.html#doegenerators

※2）メタモデル - OpenMDAO で使用できる標準機能

- Package openmdao.lib - SURROGATEMODELS
  http://openmdao.org/docs/srcdocs/packages/openmdao.lib.html#surrogatemodels

※3）最適化手法 - OpenMDAO で使用できる標準機能

- Choosing an Optimizer
  http://openmdao.org/docs/tutorials/optimization/optimizers.html