Algorithms for tuning a computer code to data

Jeong Soo Park, E-mail: jspark@jnu.ac.kr

Chonnam Natl Univ, Korea

April 9, 2015

Contents

- Introduction to code tuning
- statistical methods for code tuning
  - Approximate Nonlinear least squares method using GP metamodel
  - Bayesian approach
  - Likelihood based approach: Full Likelihood and Conditional likelihood
- Iteratively re-estimated approach: Min-Max & EM algorithm
- Toy model study
- Discussion & Summary

What is code tuning or calibration of computer model?

- Definition: The process of matching the computer model to the observed data by adjusting the unknown constants (or parameters) in the computer model.
- The unknown constants or parameters : Tuning parameters or calibration parameters
- Statistically speaking, code tuning (or calibration) is a kind of estimating the tuning parameters

Example of code tuning

- Nuclear fusion experiments: tokamak
- A simple measure of energy efficiency in tokamak is the global energy confinement time ω. The theoretical model :

$$\omega \ = \ f(\ \boldsymbol{\tau},\ P,\ I,\ N,\ B)\ , \tag{1}$$

  where f is a known function calculated by a very complex simulation code (called Baldur).
- $\boldsymbol{\tau} \ = \ (\ \tau_1\ , \tau_2,\ \tau_3,\ \tau_4\ )$ are the unknown tuning parameters which have physical meaning.
- Input Variables: P is the input total power, I is the plasma current, N is the electron density, B is the magnetic field

- Real experimental data: 42 observations from PDX tokamak
- The experimental data consisted of 4 input variables (P, I, N, B) and the real observation ω_E
- In computer code, we treat $\boldsymbol{\tau}$ as input variables ($\underline{T}$). There are eight independent variables ($\underline{T}$, P, I, N, B) and computer response ω_C.
- How can we estimate the tuning parameters?

Nonlinear least squares method for code tuning

- τ is estimated by minimizing the residual sum of squares

$$RSS(\tau) = \sum_{i=1}^{n_E} [\, y_{E\,i} - Y(\tau,\, x_{E\,i})]^2, \tag{2}$$

  where $y_{E\,i}$ is an observed response (ω_E) from real experiments and $Y(\tau,\, x_{E\,i})$ is the corresponding theoretical value (ω_C) from Baldur
- Here, we assume the computer model is valid: no bias or inadequacy problem in the computer code

- We need to use a numerical optimization routine, because the minimizer of $RSS(\tau)$ is not analytically available
- Problem: Baldur simulator needs 5 cpu minutes on a Cray supercomputer for one execution
- Computationally infeasible to minimize $RSS(\tau)$ by using a numerical optimization routine

One statistical tuning method

- Replace the computer code by a cheap metamodel or an emulator
- We use the Gaussian process (GP) model among many possible choices
- Minimizing criterion

$$RSS_P(\boldsymbol{\tau}) = \sum_{i=1}^{n_E} [\, y_{E\,i} - \hat{Y}(\boldsymbol{\tau},\ x_{E\,i})]^2, \tag{3}$$

where $\hat{Y}(\boldsymbol{\tau},\ x_{E\,i})$ is the empirical BLUP of the true computer response Y at $(\boldsymbol{\tau},\ x_{E\,i})$: Kriging

- Computation of $\hat{Y}$ is quick. It makes the problem feasible: Approximate NLS method (ANLS)

References, for example

- Park (1991) Tuning Complex Computer Codes to Data..., Thesis
- Cox, Singer and Park (2001) Statistical method for tuning complex computer code ..., Comp Stat & Data Anal
- Kennedy and O'Hagan (2001) Bayesian Calibration of Computer models, JRSS-B
- Higdon, Kennedy, Cavendish et al (2004). Combining field data and computer simulations for calibration ... SIAM J Sci Comput

References, for example

- Loeppky, Bingham, Welch (2006) Computer Model Calibration or Tuning in Practice. Tech Report, Univ of British Columbia
- Han, Santner, Rawlinson (2009). Simultaneous determination of tuning and calibration parameters .... Technometrics
- Kumar (2015) Sequential tuning of complex computer models. J Stat Comp Sim
- Many references .....

GP metamodel

- Modeling:

$$y(\underline{x}) \;=\; \sum_{j=1}^{d} \beta_j f_j(\underline{x}) \;+\; Z(\underline{x}) \;+\; e, \qquad (4)$$

  where f's are known functions and β's are regression coefficients.

- $Z(.)$ is assumed to be a Gaussian process with mean zero and covariance between $Z(\underline{t})$ and $Z(\underline{u})$,

$$V(\underline{t}, \underline{u}) \;=\; \sigma_Z^2 \; R(\underline{t}, \underline{u})$$

  where σ_Z^2 is the process variance, $R(\underline{t}, \underline{u})$ is the correlation function.

- e is assumed to be normaly distributed $N(0, \sigma_e^2)$. When the response of a computer code is deterministic, we do not include the e term in model (4).

- The covariance function we have used is from the power exponential family, among many possible choices

$$R(\underline{t}, \underline{u}) \;=\; exp\,[-\sum_{i=1}^{d} \theta_i |\; t_i - u_i|^2],\; \theta_i \geq 0 \qquad (5)$$

- GP parameters: $\psi = (\boldsymbol{\beta},\ \boldsymbol{\theta},\ \sigma_Z^2,\ \gamma_e)$, where $\gamma_e = \sigma_e^2/\sigma_Z^2$

MLE for GP parameter estimation

- Maximizing the likelihood function wrt ψ

$$L(\underline{y}; \psi, \underline{x}) = \frac{(2\pi\sigma_Z^2)^{-n/2}}{\sqrt{|V|}} exp\left(-\frac{(y - F\beta)^t V^{-1}(y - F\beta)}{2\,\sigma_Z^2}\right) \quad (6)$$

  where F is a design matrix.

- The MLE of σ^2 and $\underline{\beta}$ are given by

$$\widehat{\beta} = (F^t V^{-1} F)^{-1} F^t V^{-1} y, \qquad \widehat{\sigma_Z^2} = \frac{1}{n}(y - F\widehat{\beta})^t V^{-1}(y - F\widehat{\beta}), \quad (7)$$

- Negative log likelihood is proportional to

$$\lambda(\psi) = n\ log\ \widehat{\sigma_Z^2} + log\ |V|$$

MLE for GP parameter estimation

- Since the likelihood equations do not lead to a closed form solution, a numerical optimization procedure is required.
- One can use a R-package "DiceKriging" provided by Roustant, Ginsbourger, Deville (2012) *J. Stat. Software*

Notations: There are two data sets

- E-data: real Experimental data, $n_E,\ X_E(\boldsymbol{\tau}),\ F_E,\ y_E,\ \beta_E$
- C-data: Computer experimental data, $n_C,\ X_C(T),\ F_C,\ y_C,\ \beta_C$
- B-data: Both of (combined) E-data and C-data, $n_B = n_E + n_C$
- Tuning parameters : $\boldsymbol{\tau}$
- GP parameters : ψ : $\psi_E,\ \psi_C$

Bayesian code tuning using GP model

- Kennedy and O'Hagan (2001) considered the posterior distribution of $(\boldsymbol{\tau},\ \psi_1)$ given data and $\hat{\psi}_2$. Prior information on $(\boldsymbol{\tau}, \psi_1)$ are used.
- They considered the inadequacy term c_2 of the computer model, which was estimated by using E-data.
- They used the MLE to estimate some GP parameters (ψ_2) by using C-data
- Numerical integration was used
- Advantage: It takes account of the uncertainty of prediction $(\hat{Y})$, and model inadequacy correction was applied. This method worked better than the ANLS for their one example.

Other Bayesian approaches

- For example, Higdon, Kennedy, Cavendish et al (2004) used full Bayesian modeling and MCMC for code tuning
- Bayarri, Berger, Paulo, Sacks et al.(2007) used full Bayesian approach and MCMC for validating computer model, but is similar to Bayesian calibration
- Goldstein and Rougier (2007) used Bayes linear modeling for code tuning

Separation of tuning and calibration parameters

- Han, Santner and Rawlinson (2009) defined:
- Tuning parameters: have no meaning in the physical experiments but needed to control the solution of numerical algorithm in the computer code. For example, the amount of discretization in the curve used to describe some phenomenon in the model
- Calibration parameters: have meaning in the physical experiments, but their values are either unknown or unmeasured during the running of the physical experiments, For example, the parameters in nuclear fusion model.

Separation of tuning and calibration parameters

- They proposed a method of simultaneously determining both parameters. Basically, they estimated the tuning parameters using a kind of the ANLS, and estimated the calibration parameters using a Bayesian approach and MCMC
- In this presentation, I use the term "tuning parameter" only: no separation yet

Likelihood-based approaches for code tuning

- Since the GP model is assumed to the E-data as well as the C-data
- Cox, Singer and Park (2001) considered the likelihood function of $(\boldsymbol{\tau},\ \psi)$ for the B-data (combined data)
- Full MLE: The method maximizes the Full likelihood wrt all parameters $(\boldsymbol{\tau},\ \psi)$
- They estimated $(\boldsymbol{\tau},\ \psi)$ simultaneously using the B-data

Full likelihood function

- Negative Full log likelihood is proportional to

$$\lambda_B(\boldsymbol{\tau}, \psi) \;=\; n_B \; log \; \widehat{\sigma^2}_B \;+\; log \mid V_B \mid$$

where

$$\frac{1}{\sigma_Z^2} V_B = \begin{bmatrix} R_{CC} & R_{CE} \\ R_{EC} & R_{EE} \end{bmatrix} + \begin{bmatrix} \gamma_C I & 0 \\ 0 & \gamma_E I \end{bmatrix}, \tag{8}$$

- $$\widehat{\sigma^2}_B \;=\; \frac{1}{n_B}(y_B - F_B\widehat{\beta}_B)^t {V_B}^{-1}(y_B - F_B\widehat{\beta}_B), \tag{9}$$

- Note that F_E in F_B, R_{EC}, R_{EE} are function of $\boldsymbol{\tau}$
- It fails to decouple the tuning parameters and the GP parameters

Conditional likelihood approach

- Cox, Singer and Park (2001) proposed another one
- Maximizing the Conditional likelihood of $(\boldsymbol{\tau},\ \psi_E)$ for E-data given that C-data and $\hat{\psi}_C$.
- Conditional distribution of y_E given y_C is a normal with mean and covariance

$$\mu_{E|C} = F_E\beta_E + V_{CE}^t V_{CC}^{-1}(y_C - F_C\beta_C)$$

$$V_{E|C} = V_{EE} - V_{CE}^t V_{CC}^{-1} V_{CE}$$

Conditional likelihood approach

- Negative conditional likelihood is proportional to

$$\lambda_{E|C}(\boldsymbol{\tau},\ \psi_E|\ \hat{\psi}_C)\ =\ n_E\ log\ \widehat{\sigma^2}_{E|C}\ +\ log\ |\ V_{E|C}|$$

  where $\widehat{\sigma^2}_{E|C} = \frac{1}{n_E}(y_E - \mu_{E|C})^t {V_{E|C}}^{-1}(y_E - \mu_{E|C})$
- Names of two versions: SMLE and PMLE, according to how to estimate ψ_E
- Based on the toy-model study, they suggested to use PMLE
- Advantage: It decouples two parameters. Statistical inference based on the likelihood such as confidence region of $\boldsymbol{\tau}$ and testing hypothesis on $\boldsymbol{\tau}$ are available after code tuning

Maximum a posterior approach

- If there are some prior information on $\boldsymbol{\tau}$, the objective function for maximization is changed to

$$\textit{Likelihood} \times p(\boldsymbol{\tau})$$

- One may take two stages:

1. Estimates ψ from the C-data only by MLE without considering the prior information on $\boldsymbol{\tau}$
2. Plug the MLE into the posterior, that is, the objective function is

$$\textit{Likelihood}(\boldsymbol{\tau},\ \psi_E|\hat{\psi}_C) \times p(\boldsymbol{\tau})$$

Discussion: Issues on Code tuning

- Design issue: A careful selection of input points is required, which is a statistical design problem for tuning a complex simulation code
- Good Latin-hypercube designs are usually used
- Data-adaptive Sequential (optimal) design strategy might be useful, specially in practice: After estimating the tuning parameters, use the information to construct an efficient design for the next stage computer experiments and code tuning

Discussion: Model selection issue

- A good representative metamodel among many GP models should be selected: see Marrel et al (2007) for a systematic way of selecting a GP model
- In my study, two models were considered:
- Model 1: $y(\underline{x}) = \beta_0 + \beta_1 x_1 + ... + \beta_d x_d + Z(\underline{x})$, with one common θ in the covariance function
- Model 2: Same as Model 1 but with different θ's

- Introduction to code tuning
- statistical methods for code tuning
  - ANLS method using GP metamodel
  - Bayesian approach
  - Full Likelihood and Conditional likelihood approaches
- Iteratively re-estimated approach: Min-Max & EM algorithm
- Toy model study
- Discussion & Summary

Generalized ANLS

- Lee and Park (2014) considered some generalized ANLS methods:
- Minimizing

$$GRSS_p(\boldsymbol{\tau}) = [\ y_E - \ \hat{Y}(\boldsymbol{\tau},\ x_E)]^t W^{-1} [\ y_E - \ \hat{Y}(\boldsymbol{\tau},\ x_E)]$$

  wrt $\boldsymbol{\tau}$, where W is the covariance matrix of residual vector
- A special case is minimizing

$$WRSS_p(\boldsymbol{\tau}) = \sum_{i=1}^{n_E} w_i\ [\ y_{E\,i} - \ \hat{Y}(\boldsymbol{\tau},\ x_{E\,i})]^2$$

  where w_i are weights.
- These criterion may be useful when the residuals have some patterns like correlated or non-constant variance

Iteratively re-estimation algorithm

- In $WRSS_p$, one can get the weights w_i by using an iteratively re-weighted method as sometimes done in regression analysis
- Our idea of Iteratively re-estimation algorithm was inspired by this iteratively re-weighted LS algorithm
- Two methods we recently proposed are : Iteratively re-estimated ANLS, and EM algorithm

Iteratively re-estimated ANLS

- Step 1: It starts with ANLS, and get $\widehat{\tau}$
- Step 2 (maximization): Then find the MLE of GP parameters using the B-data in which $\widehat{\tau}$ were plugged into the E-data
- Step 3 (minimization): Using the new GP parameter estimates, minimize RSS_p to get a new $\widehat{\tau}$.
- Do this iterations (Step 2 and 3) until convergence.

Iteratively re-estimated ANLS

- The key point is that we estimate GP parameters in Step 2 using the B-data
- The likelihood is similar to the Full likelihood but $\widehat{\boldsymbol{\tau}}$ is given

$$\lambda(\psi_B; y_B, X_B|\widehat{\boldsymbol{\tau}})$$

- We also use the B-data for prediction in computing RSS_p in Step 3

$$RSS_p(\boldsymbol{\tau}) = [\ y_E - \ \hat{Y}_B(\boldsymbol{\tau},\ x_E)]^t[\ y_E - \ \hat{Y}_B(\boldsymbol{\tau},\ x_E)]$$

where $\hat{Y}_B(w_0) = f_0^t\ \hat{\underline{\beta}}_B + r_{0B}^t\ \hat{V}_B^{-1}(\underline{y}_B - F_B\hat{\underline{\beta}}_B)$ where $w_0 = (\boldsymbol{\tau},\ x_E)$.

- One can use the E-data only for this prediction

Iteratively re-estimated ANLS

- Advantage: Uses more data than ANLS, takes account of correlation between E-data and C-data, and updates the estimates of the GP and tuning parameters
- We gave a new name: Min-Max algorithm because it uses minimization and maximization iteratively (and, because IRANLS is too long and hard to pronounce)

another iterative algorithm

Expectation-Maximization (EM) algorithm:

- E-Step: Set $\widehat{\boldsymbol{\tau}}$ be the conditional expectation of $\boldsymbol{\tau}$ given the MLE of the GP parameters and the B-data (= expectation of the posterior of $\boldsymbol{\tau}$ given the MLE ($\hat{\psi}$) and the B-data)
- M-Step: Maximize likelihood of the GP parameters ψ from the B-data with the E-data in which $\widehat{\boldsymbol{\tau}}$ are plugged-in (same as the Step 2 in Min-max algorithm)
- Iterate E and M steps until convergence

EM algorithm for code tuning

- Formula for the conditional expectation of τ in E-Step on k-th iteration

$$\begin{aligned} E(\tau|\underline{y}_B;\underline{x}_B,\hat{\psi}^{(k)}) &= \int \tau \frac{A\ p(\tau;\underline{x}_B,\hat{\psi}^{(k)})}{\int A\ p(\tau;\underline{x}_B,\hat{\psi}^{(k)})\ d\tau}\ d\tau \\ &= \tau^{(k+1)} \end{aligned} \tag{10}$$

  Here $A = f(\underline{y}_B|\tau;\underline{x}_B,\hat{\psi}^{(k)}) = pdf\ \ of\ \ MN(F_B\widehat{\beta}^{(k)},\hat{\sigma}^{2(k)}V_B{}^{(k)})$, $p(\tau;\underline{x}_B,\hat{\psi}^{(k)}) = U(a,b)\ \ ,\ \ \psi^{(k)} = (\underline{\theta}^{(k)},\underline{\beta}^{(k)},\sigma^{2(k)},\gamma_E^{2(k)})$

- Note that F_B and V_B are functions of τ.
- A numerical integration method by Genz and Malik (1980) in R package "cubature" (Steven; 2009) was used.
- Prior information on τ is easily incoperated

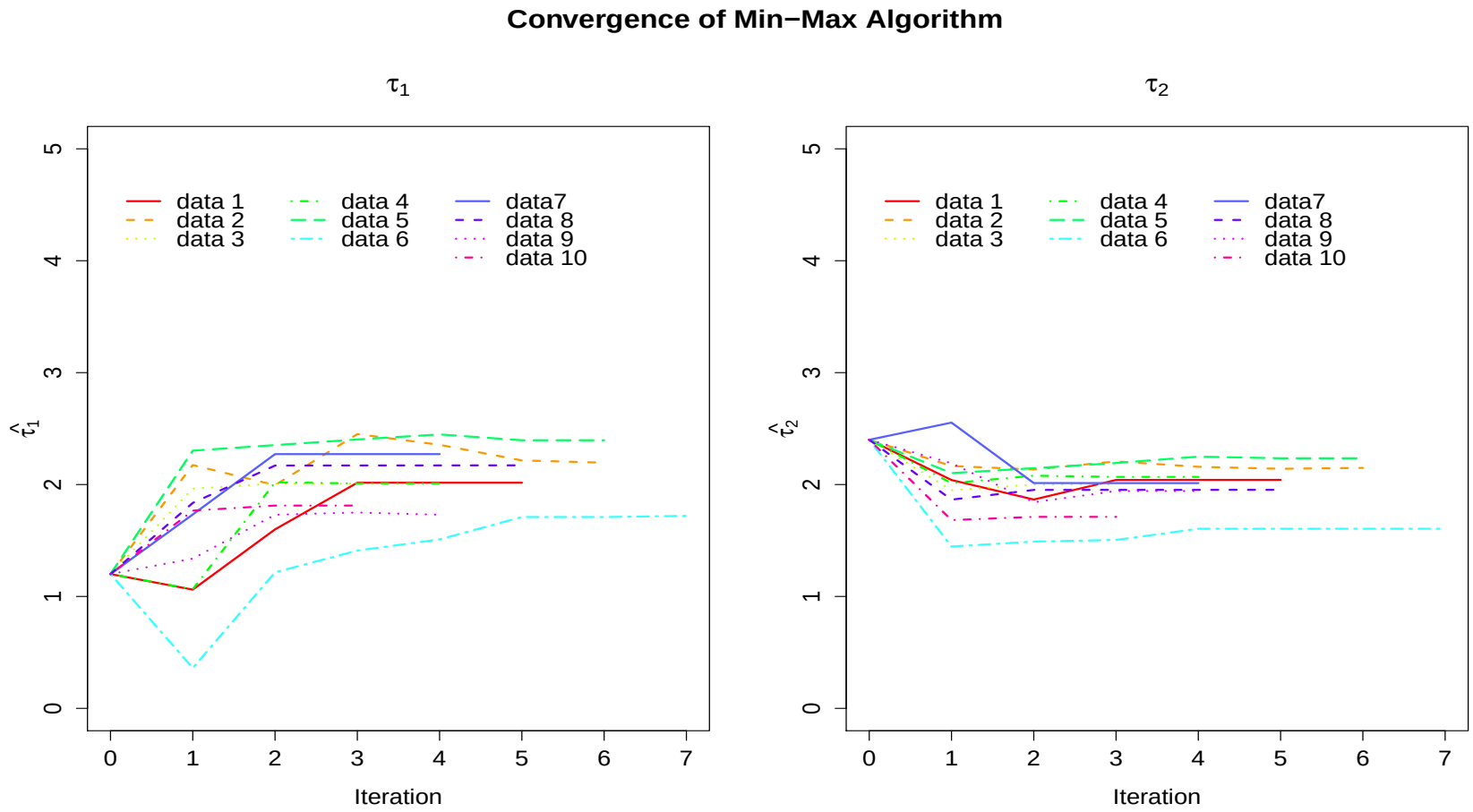

Figure: Convergence of Min-Max algorithm based on 10 trials for test function 1

Convergence of EM algorithm

Based on 10 trials for test function 1
EM algorithm stops at
Q1: 6 iterations
Median: 10 iterations
Q3: 28 iterations

Uncertainty of the estimates of tuning parameters

- To get the standard error of $\boldsymbol{\tau}$, one can appeal to the usual (increasing domain) asymptotic normality of the MLE. We can use the inverse of the observed Fisher information matrix (FIM) for the negative "unconcentrated" log-likelihood function at $\widehat{\boldsymbol{\tau}}$.
- For the approximate $100(1-\alpha)\%$ confidence region of $\boldsymbol{\tau}$,

$$\{\boldsymbol{\tau} : (\boldsymbol{\tau} - \widehat{\boldsymbol{\tau}})^t H^{-1}_{(\widehat{\boldsymbol{\tau}})} (\boldsymbol{\tau} - \widehat{\boldsymbol{\tau}}) \leq {\chi_k}^2 (1-\alpha)\}, \tag{11}$$

where $H_{(\widehat{\boldsymbol{\tau}})}$ is the principal major (corresponding to $\widehat{\boldsymbol{\tau}}$) of the inverse of the FIM at $\boldsymbol{\tau} = \widehat{\boldsymbol{\tau}}$ (k is the number of parameters in $\boldsymbol{\tau}$).

Uncertainty of the estimates of tuning parameters

- When using the ANLS method, we rely on the asymptotic theory for nonlinear least squares estimator (Draper and Smith, 1980). The approximate $100(1-\alpha)\%$ confidence region of $\boldsymbol{\tau}$ is obtained by

$$\{\boldsymbol{\tau} : RSS_P(\boldsymbol{\tau}) \leq RSS_P(\widehat{\boldsymbol{\tau}})[1 + \frac{k}{n_E - k} F_\alpha(k, n_E - k)]\}, \qquad (12)$$

- For any tuning methods, One can use a resampling technique such as a parametric bootstrap, by using random sampling from the GP models

A classification of code tuning algorithms

Method	τ estimation	Model building	Outer iteration
NLS	NLS(E)	Sim Code	1
ANLS	ANLS(E)	MLE(C)	1
FMLE	MLE(C, E)		1
SMLE	MLE(E|C)	MLE(C)	1
EM	Cond Expect(C, E)	MLE(C, E)	several times
KOH BC	Cond Expect(E)	MLE(C)	1
Min-Max	ANLS(C, E)	MLE(C, E)	several times

EM algorithm here can be viewed as an iteratively re-estimated version of KOH BC.

Toy model study

Test functions to compare the performance of methods

- 5 test functions are tried
- One example: test function 1

$$Y(\tau, x) = \tau_1 \exp(\tau_2 + x_1) + \tau_1 x_2^2 - \tau_2 x_3^2 \tag{13}$$

$y_E = Y(\tau, x) + e$, $n_E = n_C = 30$
E-Data: $\tau_1 = 2, \tau_2 = 2$, $x_1 \sim U(-3, 3), x_2 \sim U(-3, 3)$,
$x_3 \sim U(0, 6)$, $e \sim N(0, 1)$
C-Data: $T_1 \sim U(0, 5), T_2 \sim U(0, 4)$, x_1, x_2, x_3: same as E-Data

- Other 4 test functions are omitted
- For each test function, 30 Latin-hypercube designs were tried to get 30 estimates so that we can check the variations of the methods

MSE comparison from 5 test functions

Table: MSE of tuning methods for 5 test functions

method	Test1	Test2	Test3	Test4	Test5
ANLSE 1	0.381	0.615	1.866	2.460	0.222
ANLSE 2	0.746	0.606	3.591	2.721	0.208
SMLE 1	0.349	0.528	2.291	2.453	0.190
SMLE 2	0.733	0.781	4.191	4.154	0.447
EM 1	0.163	**0.235**	1.060	**0.707**	0.065
EM 2	**0.152**	0.267	**1.014**	0.952	**0.057**
Min-max1	0.211	0.577	1.591	1.818	0.265
Min-max2	1.005	0.493	3.013	2.280	0.190

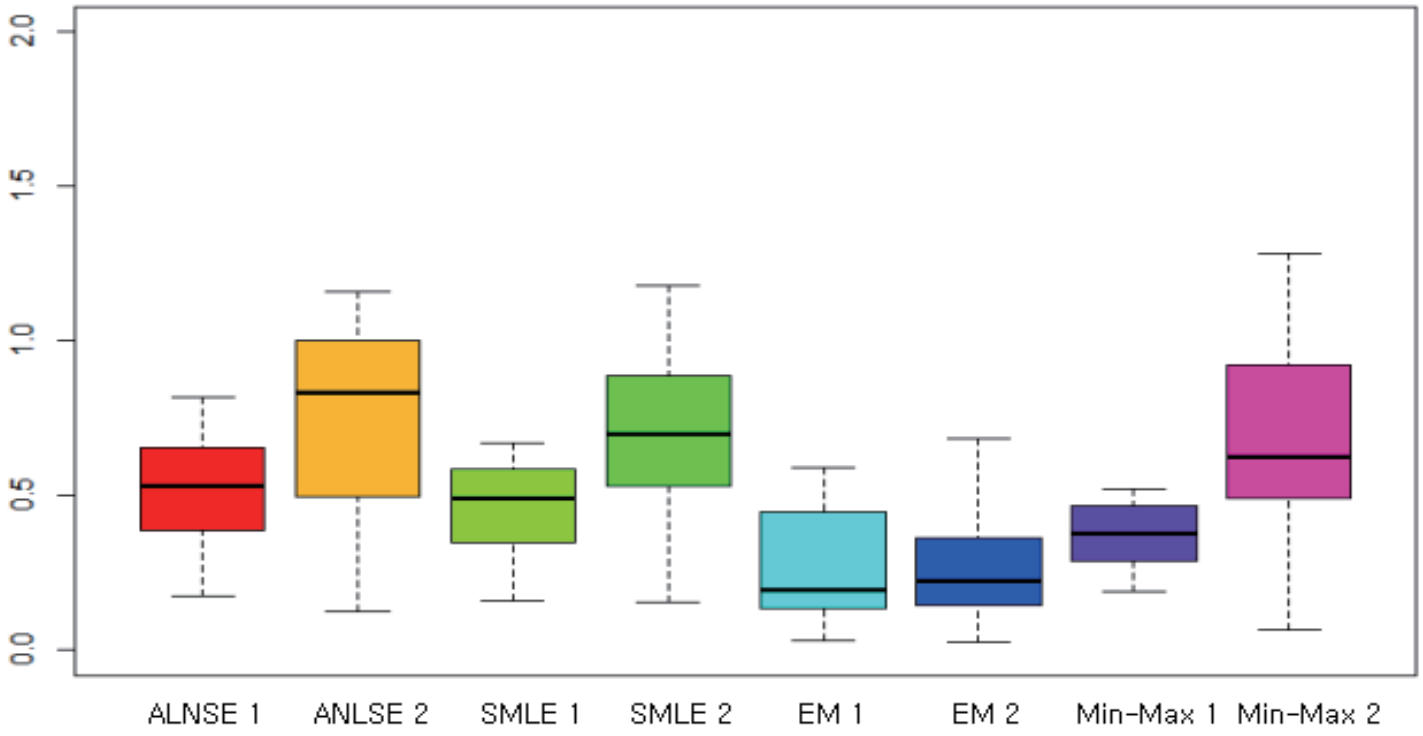

Figure: Box plot of distance to the true value in test function 1.

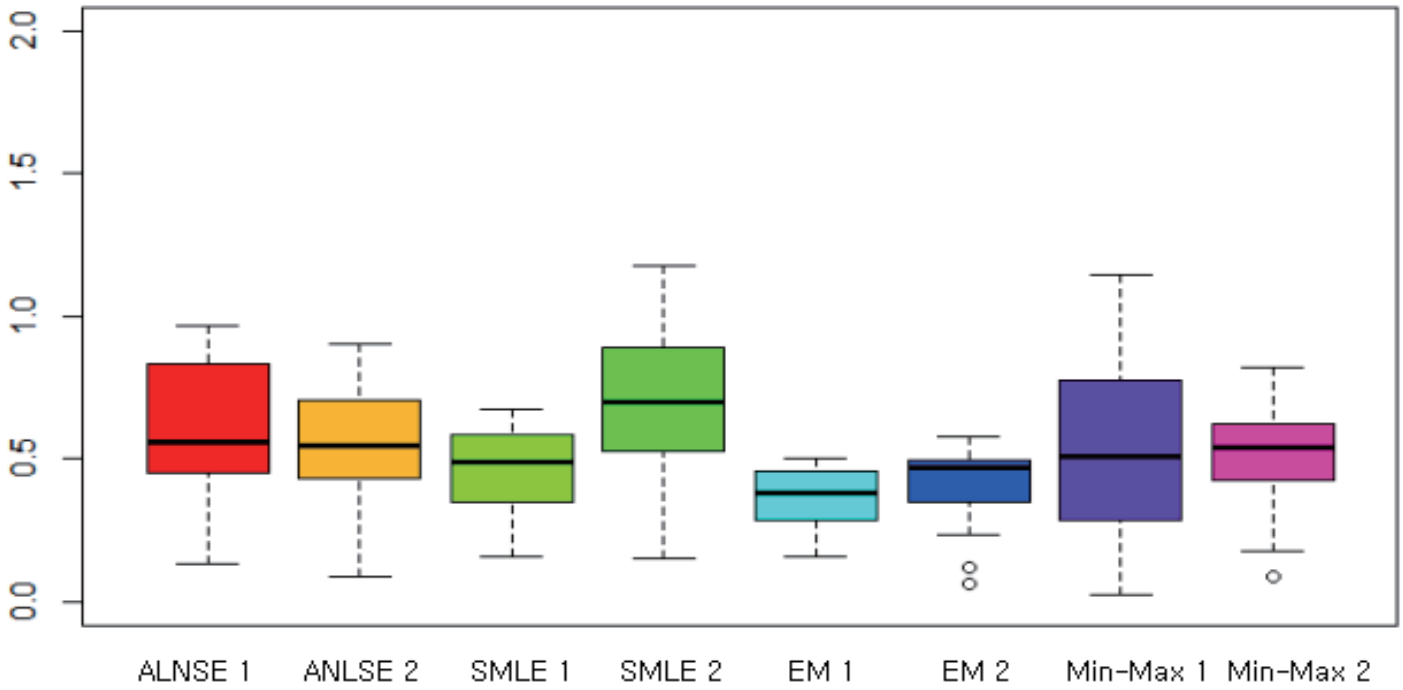

Figure: Box plot of distance to the true value in test function 2.

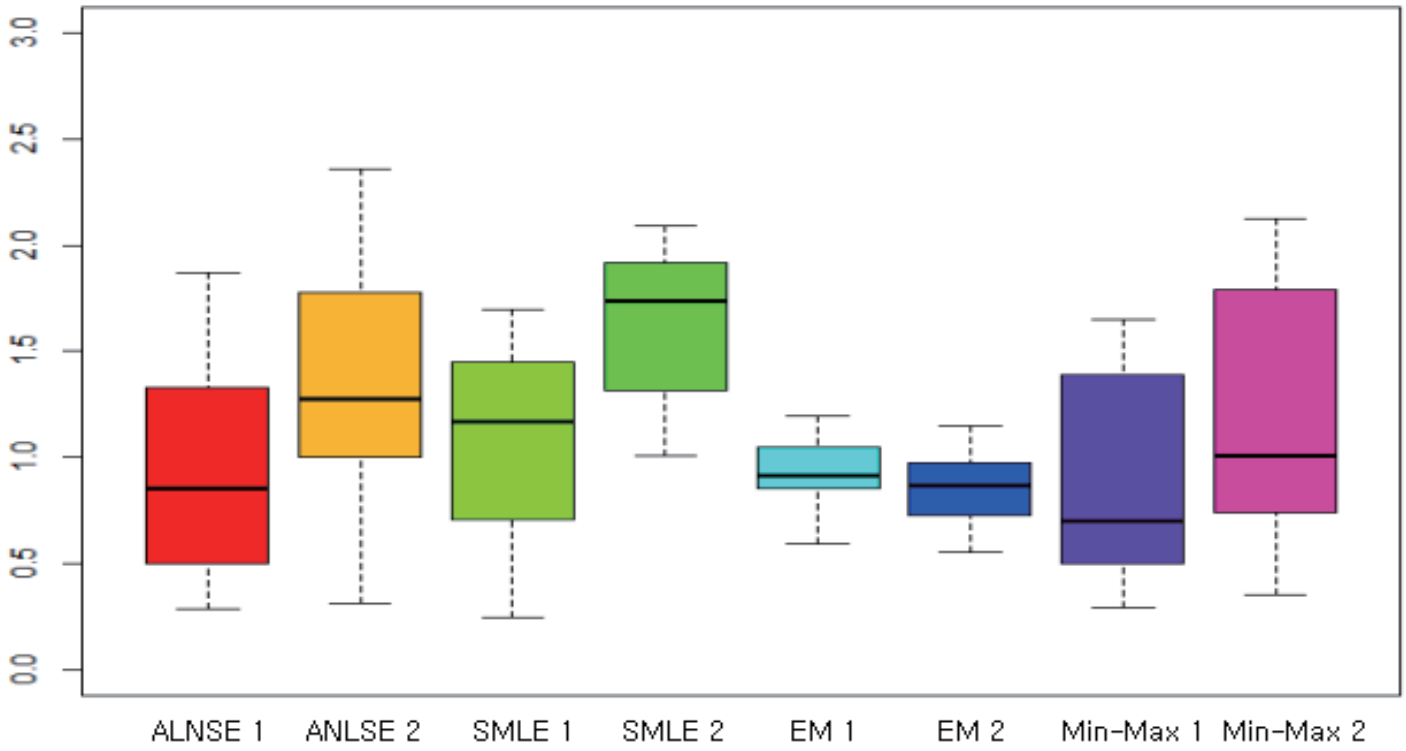

Figure: Box plot of distance to the true value in test function 3.

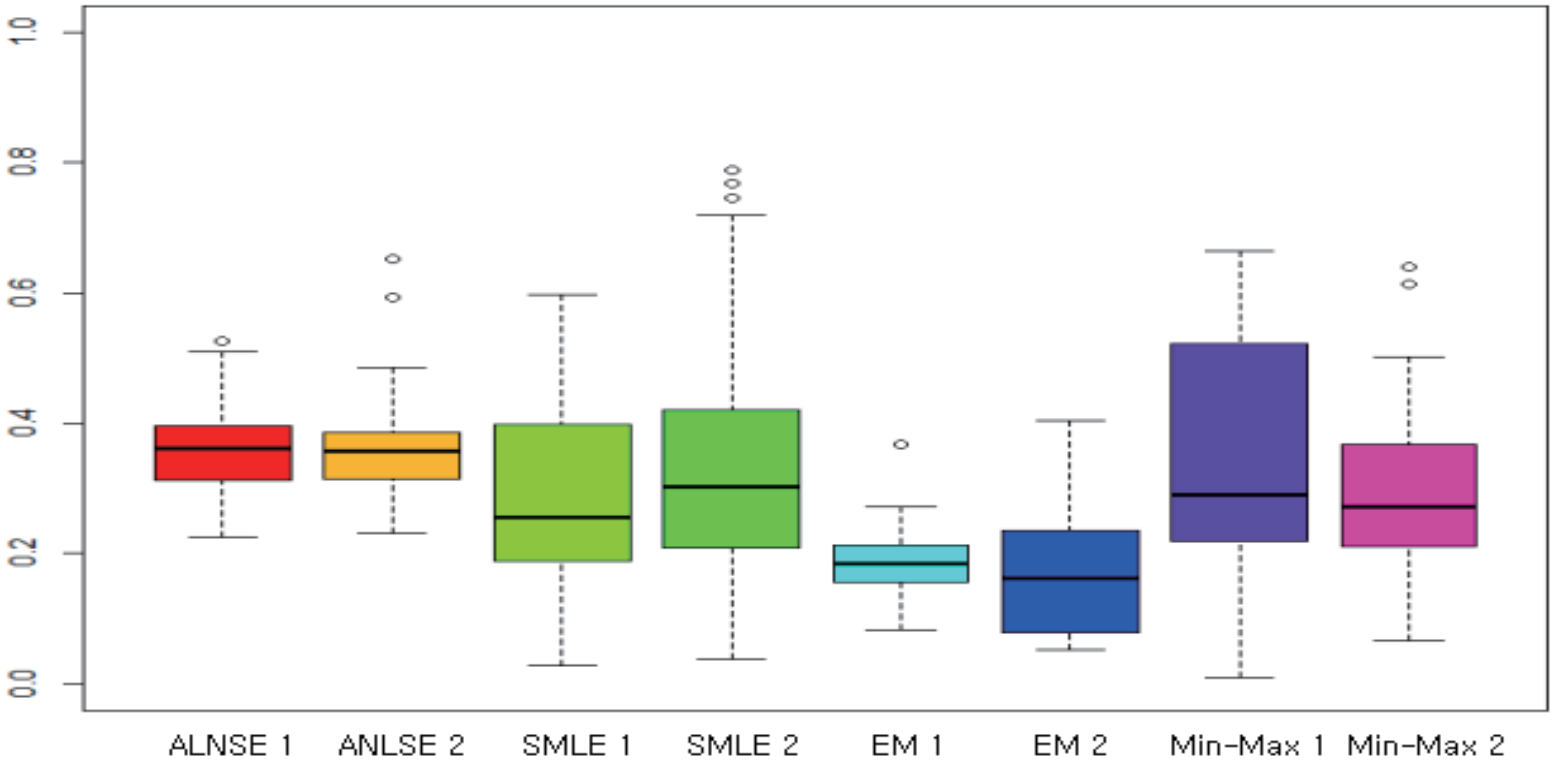

Figure: Box plot of distance to the true value in test function 4.

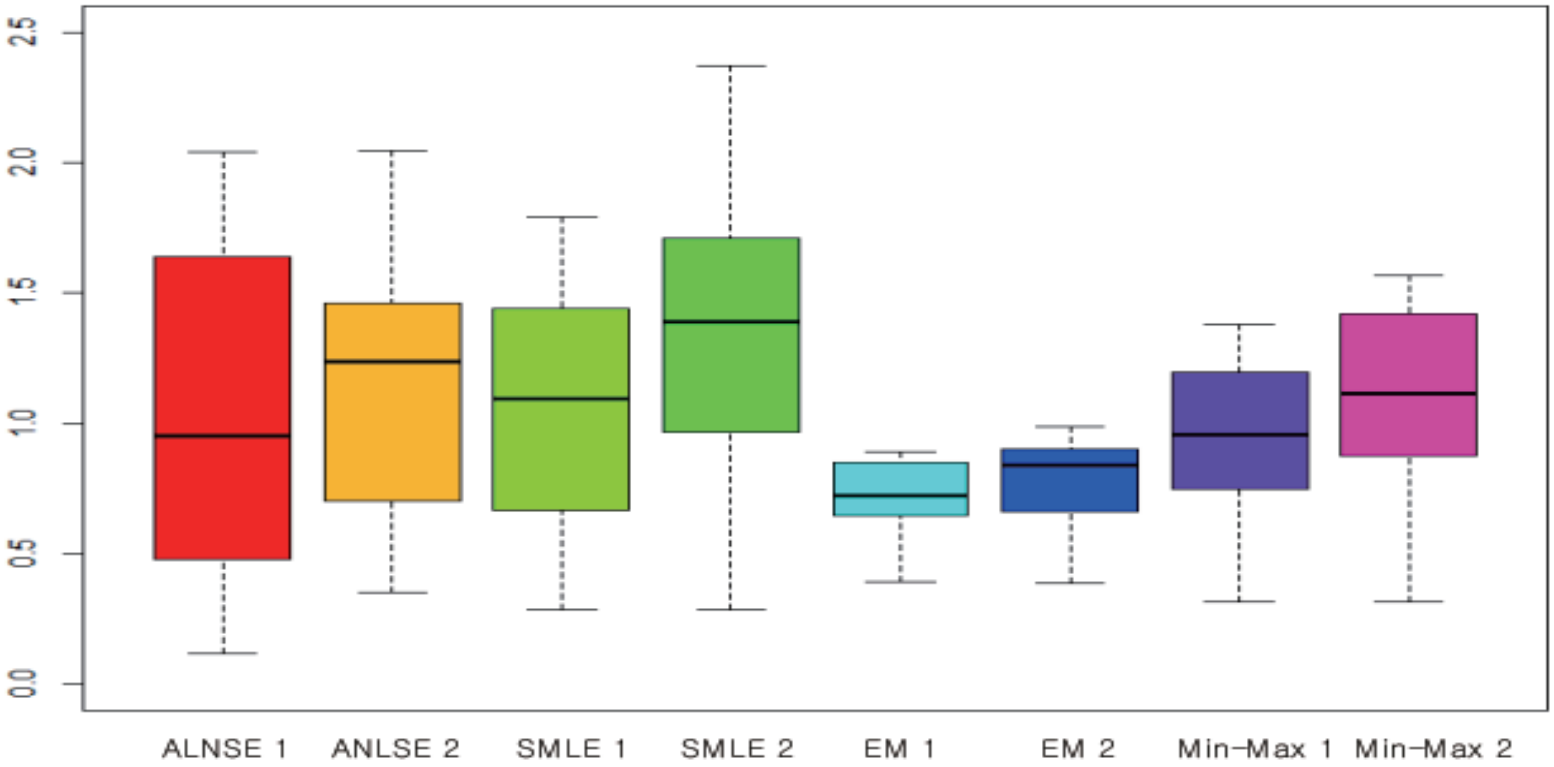

Figure: Box plot of distance to the true value in test function 5.

Result of the test function study

- EM algorithm works better than other methods
- Min-max works better than ANLS
- Why EM algorithm works better than others, specially than the Min-max?
- My answer: 1. EM uses the combined data, and iteratively updates the GP and tuning parameters.
  2. Numerical integration in computing the conditional expectation may be more stable than the numerical minimization routine in least squares method
  3. what else? EM algorithm here is viewed as an iteratively re-estimated version of KOH BC

Computing time for tuning methods

Table: Averaged computing time in second for tuning methods, for $n_E = n_C = 30$, (from 30 trials for each test function)

(unit: sec)

function	No. of τ	No. of X	ANLS	SMLE	Min-max	EM
Test 2	3	4	149	150	661	471
Test 4	4	6	185	295	1207	1300
Test 5	2	6	23	117	435	320

Discussion: Limitation and future study

- We had a limitation of extending our test function study to higher dimensional τ, because of heavy computing time in Min-max and EM algorithm
- Numerical integration for more than 5 (or 10) dimensions in EM algorithm may not be practical. Monte Carlo approach may be useful.
- We may need to study more in quantifying the uncertainty of the estimates of tuning parameters which take account of the uncertainty of the estimated GP models

Discussion: Limitation and future study

- Some tuning methods are heavily dependent on the performance of numerical optimization algorithm. We need better practice in using the optimization routine.
- Performance study of the code tuning methods is very dependent on the form of test functions. The building of a test bed (a set of test functions) for code tuning purpose is demanded for future research

Discussion: future study

Iteratively re-estimated version of the conditional MLE is possible:

- Step 1: Get $\widehat{\boldsymbol{\tau}}$ by maximizing the Cond Likelihood of $\boldsymbol{\tau}$ on the E-data given $(y_C, \hat{\psi}_C)$
- Step 2: (maximization) Get $\hat{\psi}_B$ from the B-data in which $\widehat{\boldsymbol{\tau}}$ are plugged-in the E-data
- Step 3: (maximization) Get $\widehat{\boldsymbol{\tau}}$ by maximizing the Cond Likelihood of $\boldsymbol{\tau}$ on the E-data given $(y_C, \hat{\psi}_B)$
- Iterate Step 2 and 3 until convergence: "Max-Max algorithm"

Discussion: future study

I think that KOH BC may be extended to an iteratively re-estimated version in a different way, or similar way, to ours.

Discussion: future study

- Sequential (optimal) designs for efficient code tuning is important and be a topic of future study
- Stage 1: Good design, computer experiments, model building and code tuning
- Stage 2: Using the information in Stage 1, construct a good updating design which may minimizes the uncertainty of $\boldsymbol{\tau}$. For example, find a design that minimizes the entropy of $\boldsymbol{\tau}$ given $(D_1, \widehat{\boldsymbol{\tau}}_1)$. The design region is reduced by knowing $\widehat{\boldsymbol{\tau}}$.
- Stage 3: Continue until stopping rule is satisfied. The Max MSEP can be used for the Stopping rule.

Summary

- Algorithms for code tuning methods are briefly reviewed
- Iteratively re-estimated algorithms are considered
- EM algorithm turns out to be the best in toy model study
- Statistical issues for code tuning are discussed

Thank You