# Empirical Study of Optimal Prediction of Traveling Salesman Problem using Gaussian Process Regression

WanatchapongKongkaew* and JutaPichitlamken
*Department of Industrial Engineering, Faculty of Engineering,*
*Kasetsart University, Bangkok10900, Thailand*
*wanatchapong.k@psu.ac.th, juta.p@ku.ac.th

## Abstract

We propose a new method of predicting the optimal tour given the arbitrary sample tours. We apply the theory of Gaussian process regression (GPR) to a deterministic traveling salesman problem (TSP) with a single salesman. The problem is formulated as a Gaussian process regressionmodel; the target variable is the total distance of traveling tour while the predictor is the traveling tours. In the experiment, we implement this method forthree test problems with different number of cities, and compare its performance in terms of optimal solutions.The computational timeis also considered. As an empirical result, our proposed method can be applied to solve the TSP and give the total distance of predicted tours nearoptimal solutions, roughly30-40% away from the optimal solution. Moreover, the experiment indicates that the inputs of GPR, the initial hyperparameters and the sample size of tours,affect the solution quality. The algorithmconsumes reasonable computational time becauseit uses very small sample size of tours,comparing with all possible tours.

***Keywords***:Algorithm, Gaussian process regression, Traveling salesman problem.

## 1. Introduction

The traveling salesman problem is one of the well-known *NP-hard* problems in the field of combinatorial optimization. The objective of the TSP problem is to find the shortest Hamiltonian cycle among $n$ cities, where the salesman visit each of the $n$ cities exactly once and then return to the starting city. Although its mathematical formulation is simple, the TSP is complex and it has been continually challenged by mathematicians throughout centuries. TSP applications can be found in many fields including the drilling of printed circuits boards [1], transportation and logistics problem [2], material handling in a warehouse [3], genome rearrangement [4], or job sequencing on a single machine or assignment problems [5].

There are several algorithms designed to solve the TSP problem. The exact algorithm would be to try all possible permutations (order combinations), but it uses brute force and takes more time to find the optimal solution than the branch and bound method [6] or the cutting plane method [7]. On the other hand, various heuristic and approximation algorithms, e.g., nearest neighboralgorithm (or the so-called greedy algorithm) [8], Lin-Kernighan heuristics [9, 10], *k*-opt heuristic [11] can quickly find solutions that are near the optimal solution for extremely large problem on Euclidean distances ranging from 10,000 to 10,000,000 cities. The randomized approaches (genetic algorithm [12, 13], tabu search [14], ant colony optimization [15], particle swarm optimization [16], simulated annealing algorithm [17], and the cross entropy method [18]) can find a route extremely close to the optimal route.

In this paper, we propose a new method of predicting the optimal tour. We assume that the sample tours are given. We calculate the total distance of each sample tour, and then we use the sample tours and their total distance to predict the optimal traveling tour and its total distance using Gaussian process regression.

This paper is organized as follow. In Section 2, we introduce Gaussian process regression and review its applications. In Section 3, we describea conceptual framework of this approach. Next, we present our probabilistic method for solving the TSP. In Section 4, we discuss the experimental results for different test problem. Lastly, we conclude in Section 5.

## 2. Gaussian Process Regression

The Gaussian process regression (GPR) is known as a probabilistic approach for a regression model due to its practical and theoretical simplicity and excellent generalization ability [19]. For applications, GPR is applied in many fields, particularly in machine learning, e.g., to estimate the depth of a point in space from observing its image position in two different cameras [20], to find near optimal sensor placements in the task as an instance of the art-gallery problem [21], to assess a user preference function for automatically generating music playlists [22], to learn motion and observation non-parametric system models for sequential state estimation and to apply its algorithm to the problem of tracking an autonomous micro-blimp [23]. For traffic problems, GPR is applied to predict the traveling time for an arbitrary path and tested with realistic traffic data of downtown Kyoto in Japan [24].

## 3. Approach and Algorithm

We describe how to apply the GPR model for estimating the optimal tour when some sample tours aregiven. The notations in this paper are as followed: vectors are represented by bold face, such as $\mathbf{x}_s$, and all ones are assumed to be column vectors. The transpose of vectors or matrices is denoted by $^{\mathrm{T}}$. Matrices are represented by italicized Roman symbols with bold face such as $\boldsymbol{K}$, and the $(i, j)$ element of a matrix is represented by $\boldsymbol{K}_{i,j}$. The identity matrix isrepresented as $\boldsymbol{I}$.

### *3.1 Standard Linear Regression Model*

We have a training set $\boldsymbol{T}_s$ of $n$ observations, $\boldsymbol{T}_s = \{(x_i, y_i) \mid i = 1,\ldots,n\}$, where $\mathbf{x}$ denotes an input vector (traveling tour) of $n_c$ cities and $\mathbf{y}$ denotes an output

vector of total distance of traveling tours [19]. The column vector inputs for all $n$ observations are aggregated into the $n_c \times n$ matrix of $\boldsymbol{X}$, and the total distance in each traveling tour are aggregated in the column vector $\mathbf{y}$, so we can write $\boldsymbol{T}_s = (\boldsymbol{X}, \mathbf{y})$. For review of the Bayesian analysis, the standard linear regression model with Gaussian noise is $y = f(\mathbf{x}) + \varepsilon$ and $f(\mathbf{x}) = \mathbf{x}^{\mathrm{T}}\mathbf{w}$ ,where $\mathbf{x}$ is the input vector, and $\mathbf{w}$ is a vector of weight (parameters) of a linear model or offset [19]. We assume that the observed values $y$ differs from the function value $f(\mathbf{x})$ by an additive noise $\varepsilon$, which follows an independent and identically distributed Gaussian distribution with zero mean and variance $\sigma_n^2$, that is $\varepsilon \sim N(0, \sigma_n^2)$, where $n$ is the number of observations.

For Gaussian distribution, the probability density of the observations given the parameters which is estimated over all cases in a training set is

$$p(\mathbf{y} \mid \boldsymbol{X}, \mathbf{w}) = \prod_{i=1}^{n} p(y_i \mid \mathbf{x}_i, \mathbf{w}_i) \sim N(\boldsymbol{X}^{\mathrm{T}}\mathbf{w}, \sigma_n^2 \boldsymbol{I}),$$

where $\mathbf{w}$ is a bias weight or offset with a zero mean Gaussian prior with covariance matrix $\boldsymbol{\Sigma}_p$, $\mathbf{w} \sim N(0, \boldsymbol{\Sigma}_p)$. The posterior distribution over the weights based on the Bayesian linear model is computed by Bayes' rule. We obtain the form of posterior Gaussian distribution with mean $\bar{\mathbf{w}}$ and covariance matrix $\boldsymbol{A}^{-1}$ as

$$p(\mathbf{w} \mid \boldsymbol{X}, \mathbf{y}) \sim N\left(\bar{\mathbf{w}} = \frac{1}{\sigma_n^2}\boldsymbol{A}^{-1}\boldsymbol{X}\mathbf{y}, \boldsymbol{A}^{-1}\right), \quad (1)$$

where $\boldsymbol{A} = \sigma_n^{-2}\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}} + \boldsymbol{\Sigma}_p^{-1}$. To make prediction for a test case, all possible parameter values are averaged and weighted by posterior probability. The predictive distribution for the function value $f_s \sim f(\mathbf{x}_s)$ at $\mathbf{x}_s$, is given by averaging the output of all possible linear models. Thus the Gaussian posterior is written as

$$p(f_s \mid \mathbf{x}_s, \boldsymbol{X}, \mathbf{y}) \sim N\left(\frac{1}{\sigma_n^2}\mathbf{x}_s^{\mathrm{T}}\boldsymbol{A}^{-1}\boldsymbol{X}\mathbf{y}, \mathbf{x}_s^{\mathrm{T}}\boldsymbol{A}^{-1}\mathbf{x}_s\right). \quad (2)$$

The predictive distribution in (2) is also Gaussian distribution, with a posterior mean of the weights from (1) multiplied by the all possible values $\mathbf{x}_s$ in a test case. Moreover, the predictive variance is a quadratic form of all possible values $\mathbf{x}_s$ in a test case multiplied with the posterior covariance matrix.

### 3.2Gaussian Process Regression Model

Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribution, by definition, and it is specified by its mean functionand covariance function [19].The mean function $m(\mathbf{x})$ and covariance function $k(\mathbf{x}, \mathbf{x}')$ are defined a real process as $m(\mathbf{x}) = \mathrm{E}[f(\mathbf{x})] = 0$ and $k(\mathbf{x}, \mathbf{x}') = \mathrm{E}[(f(\mathbf{x}) - m(\mathbf{x}))(f(\mathbf{x}') - m(\mathbf{x}'))]$. Thus, we can write the Gaussian process as $f(\mathbf{x}) = GP[m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')]$.

The prior on the noisy observations, independent and identically distributed Gaussian noise $\varepsilon$ with variance $\sigma_n^2$, becomes $\mathrm{cov}(\mathbf{y}) = \boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) + \sigma_n^2\boldsymbol{I}$. In addition, the joint distribution of the observed target values and function values at the test locations as

$$\begin{bmatrix}\mathbf{y} \\ \mathbf{f}_s\end{bmatrix} \sim N\left(0, \begin{bmatrix}\boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}) + \sigma_n^2\boldsymbol{I} & \boldsymbol{K}(\boldsymbol{X}, \boldsymbol{X}_s) \\ \boldsymbol{K}(\boldsymbol{X}_s, \boldsymbol{X}) & \boldsymbol{K}(\boldsymbol{X}_s, \boldsymbol{X}_s)\end{bmatrix}\right).$$

The predictive distribution for Gaussian process regression is $p(\mathbf{f}_s \mid \boldsymbol{X}, \mathbf{y}, \boldsymbol{X}_s) \sim N(\bar{\mathbf{f}}_s, \mathrm{cov}(\mathbf{f}_s))$, where $\bar{\mathbf{f}}_s$ is mean of function values, and $\mathrm{cov}(\mathbf{f}_s)$ is the predictive variance function values (corresponding to test input $\boldsymbol{X}_s$). For a single test point $\mathbf{x}_s$, the predictive mean and the predictive variance are

$$\boldsymbol{\alpha} = \left(\boldsymbol{K} + \sigma_n^2\boldsymbol{I}\right)^{-1}\mathbf{y}, \quad (3)$$

$$\bar{f}_s = \mathbf{k}_s^{\mathrm{T}}\boldsymbol{\alpha}, \quad (4)$$

$$V[f_s] = k(\mathbf{x}_s, \mathbf{x}_s) - \mathbf{k}_s^{\mathrm{T}}\left(\boldsymbol{K} + \sigma_n^2\boldsymbol{I}\right)^{-1}\mathbf{k}_s. \quad (5)$$

The log marginal likelihood under the Gaussian process model is

$$\log p(\mathbf{y} \mid \boldsymbol{X}) = -\frac{1}{2}\mathbf{y}^{\mathrm{T}}\boldsymbol{\alpha} - \frac{1}{2}\log\left|\boldsymbol{K} + \sigma_n^2\boldsymbol{I}\right| - \frac{n}{2}\log(2\pi). \quad (6)$$

Moreover, the graphical model for Gaussian process regression can be shown in Fig.1.

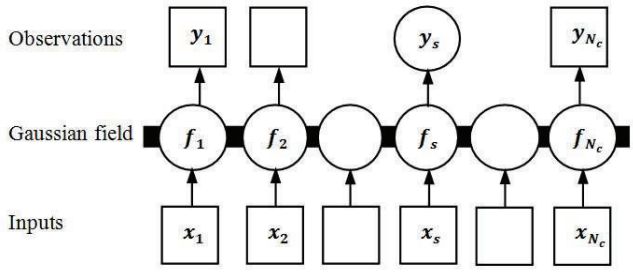


Fig. 1 Graphical model for Gaussian process regression [19]

In many practical applications, all aspects of the covariance function cannot be specified with confidence. Further, it is essential to develop the method of cross-validation for model selection. At this point, the log marginal likelihood in (6) is required to determine the hyperparameters, e.g., length-scale, for model selection.

In section 3.1, we describe a linear model with a $n_c$ city input vector $\mathbf{x}$. In this section, we assume that the basis function $\phi(\mathbf{x})$ which maps a $n_c$ city input vector $\mathbf{x}$ into $N_c$ -dimension feature space. Let the matrix $\Phi(\boldsymbol{X})$ be the aggregation of columns $\phi(\mathbf{x})$ for all cases in training data set, we obtain $f(\mathbf{x}) = \phi(\mathbf{x})^{\mathrm{T}}\mathbf{w}$, where the vector of parameters has length $n \times n_c$. Hence, the predictive distribution becomes

$$p\left(f_s \mid \mathbf{x}_s, \boldsymbol{X}, \mathbf{y}\right) \sim N\left(\frac{1}{\sigma_n^2}\phi_s^{\mathrm{T}} \boldsymbol{A}^{-1}\Phi\mathbf{y}, \phi_s^{\mathrm{T}} \boldsymbol{A}^{-1}\phi_s\right), \quad (7)$$

with $\Phi = \Phi\left(\boldsymbol{X}\right)$, $\phi_s = \phi\left(\mathbf{x}_s\right)$, and $\boldsymbol{A} = \sigma_n^{-2}\Phi\Phi^{\mathrm{T}} + \sum_p^{-1}$. In the right-hand-side term of (7), we need to invert the $\boldsymbol{A}$ matrix of size $N_c \times N_c$ to make predictions, which may not convenient if $N_c$ -dimension feature space is large. However, we can rewrite this term as

$$N\left(\begin{array}{l}\phi_s^{\mathrm{T}} \sum_p \Phi\left(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I}\right)^{-1} \mathbf{y}, \\ \phi_s^{\mathrm{T}} \sum_p \phi_s - \phi_s^{\mathrm{T}} \sum_p \Phi\left(\boldsymbol{K} + \sigma_n^2 \boldsymbol{I}\right)^{-1} \Phi^{\mathrm{T}} \sum_p \phi_s\end{array}\right), \quad (8)$$

where the covariance matrix $\boldsymbol{K} = \Phi^{\mathrm{T}} \sum_p \Phi$.

Notice that in (8) the entries of matrices, in the form of $\Phi^{\mathrm{T}} \sum_p \Phi$, $\phi_s^{\mathrm{T}} \sum_p \Phi$, or $\phi_s^{\mathrm{T}} \sum_p \phi_s$, are the form $\phi\left(\mathbf{x}\right)^{\mathrm{T}} \sum_p \phi\left(\mathbf{x}'\right)$ where $\mathbf{x}$ and $\mathbf{x}'$ are the training set and test set, respectively. Let us define $k\left(\mathbf{x}, \mathbf{x}'\right) = \phi\left(\mathbf{x}\right)^{\mathrm{T}} \sum_p \phi\left(\mathbf{x}'\right)$, and we called $k\left(\mathbf{x}, \mathbf{x}'\right)$ that a *covariance function* or *kernel*.

### 3.3Tour Construction and Representation

The sample tours are used as an input of GPR. In this section, we first explain how to construct the sample tour, and its representation is briefly described in the next paragraph.The sample tours are constructed by permuting the cities' number from 1 to $n$ (among $n$ cities) using the function "randperms" in MATLAB[25], and then they are represented as an input. There are many different representations proposed to represent a tour; for example, path representation, binary string representation, binary matrix representation, edge recombination crossover in binary representation. The most natural representation of one tour used to solve TSP is represented by path representation [26].

The path representation represents a tour as a list of $n$ cities, which should be visited and are put in order, i.e., if city $i$ is the $j$-th element of the list, then the city $i$ is the $j$-th city visited by salesman. For instance, the path representation $\left(6\ 4\ 2\ 1\ 5\ 3\right)$ is referred to a tour $6 \rightarrow 4 \rightarrow 2 \rightarrow 1 \rightarrow 5 \rightarrow 3$ [26]. However, this method seems to be not suitable for GPR becauseits representation contradicts with thetheoretical assumption, i.e., the input is formulated as a function. Hence, in this paper, the binary string representation is used to encode a tour for GPR input. Each city in a tour is encoded as a string of $\left\lceil \log_2 n \right\rceil$ bits and then a complete tour becomes a string of $n\left\lceil \log_2 n \right\rceil$ bits, e.g., a tour $4 \rightarrow 2 \rightarrow 1 \rightarrow 3$ is represented by $\left(011\ 001\ 000\ 010\right)$.

### 3.4 Proposed GPR Algorithm

We explain our GPR algorithm for a deterministic TSP with a single salesman, divided into three phases: training phase, test phase, and output phase.

**Training phase**:
We prepare the distance matrix of the training data set and determine the output of this phase.

*Input*: distance matrix $\boldsymbol{D}$.
*Algorithm*:

- Find the row vector of tours (in cities' number) using a permute function (randperms) in MATLAB.
- Aggregate all row vectors of tours into the matrix of tours, $\boldsymbol{X}$.
- Calculate the vector of total distance $\mathbf{y}$, index the distance of each pair of city in $\boldsymbol{D}$ and sum all values of them.
- Encode the row vector of tours to the binary row vector of tours.
- Find a row vector of test tour (in cities' number) using a permutation function.
- Encode a row vector of test tour to the binary row vector of test tour $\mathbf{x}_s$.

*Output*: the vector of total distance $\mathbf{y}$, the binary matrix of tours $\boldsymbol{X}$ the binary row vector of test tour $\mathbf{x}_s$.

**Test phase**:
We precompute the matrix inversion required in (3), (4), by the Cholesky factorization for the predictive total distance and the predictive variance.

*Input*: the vector of total distance $\mathbf{y}$, the binary matrix of tours $\boldsymbol{X}$,the binary row vector of test tour $\mathbf{x}_s$, covariance function $k$,initial hyperparameters, noise level $\sigma_n^2$.
*Algorithm*:

- Compute $\boldsymbol{\alpha}$ as specified in (3).
- Compute the predictive mean as specified in (4).
- Compute the predictive variance as specified in (5).
- Compute and minimizethe log marginal likelihood for GPR as specified in (6).
- Return the predictive mean $\overline{f}_s$, the predictive variance $V\left[f_s\right]$, and the log marginal likelihood, $\log\ p\left(\mathbf{y} \mid \boldsymbol{X}\right)$.

*Output*: the predictive mean $\overline{f}_s$.

**Output phase**:
We transform the predictive mean $\overline{f}_s$ to the predictive tour and display the resulting tour as follow.

*Input*: the predictive mean $\overline{f}_s$.

*Algorithm*:

- Transform the predictive mean $\overline{f}_s$ to the binary row vector of predictive tour.

- Decode the binary row vector of predictive tour to the row vector of the predictive tour (in cities' number).
- Verify the legal tour; if the obtained tour is illegal tour (all cities are not connected together into a tour), the tour improvement procedure is required [27].
- Calculate the total distance, index the distance of each pair of city in $\boldsymbol{D}$ and sum all values of them.

*Output*: the predictive optimal tour and its total distance.

In output phase, the legal tour verification and tour improvement procedure are not embedded into our GPR algorithm. However, they are imbedded into the algorithm in the further research.

## 4. Experimental Results

We implement the GPR algorithm described in Section 3 in MATLAB and test it with three different problems: thefirst one isan example in textbook [28] and others are the TSP instances from the TSPLIB95 library [29],as follow:

- Wolverine manufacturing problem: 9 cities with $3.63\times10^{5}$ of possible solutions [28].
- gr17: 17 cities with $3.56\times10^{14}$ of possible solutions [29].
- fri26: 26 cities with $4.03\times10^{26}$ of possible solutions [29].

In our experimental setting, we vary an initial length-scale ($\ell$): 1, 2, 3, and 4. In addition, the number of sample tours that are generated as an input $\boldsymbol{X}$ are 5, 10, 100, 1000 tours. We test on PC running Intel(R) Core$^{TM}$ i7-720QM 1.60 GHz processor with 4GB of memory.

We perform $n_M$ macroreplications, which each macro-replication consists of $n_m$ microreplications or searches [30].For each combination of our experiment, the macroreplication run ($n_M$) is 3, each macroreplication consists of the average of $n_m$ = 3 microreplications. Then the average value within each macroreplicationis treated as output value of GPR algorithm. Furthermore, the performance of GPR algorithm measures how the predictive value of total distance is close to an optimal solution, and the computational time is also considered.

The results of each test problem (in unit) and its optimal solutionare shown in Table 1. The plots of the average percentage of difference between the total distance of solutions and its optimal solution for Wolverine manufacturing, gr17, fri26problems areprovided in Figs. 2, 3, and 4, respectively.The computational time (in second) for solving among three problems with length-scale $\ell$ = 2 is illustrated in Fig. 5.

Table 1: Average predictive total distance (unit) among three problemsobtained by GPR algorithm

| Initial Length-scale | Wolverine Manufacturing (Optimal Solution = 46.49) | | | | gr17 (Optimal Solution = 2085) | | | | fri26 (Optimal Solution = 937) | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Sample Size (tours) | | | | Sample Size (tours) | | | | Sample Size (tours) | | | |
| | 5 | 10 | 100 | 1000 | 5 | 10 | 100 | 1000 | 5 | 10 | 100 | 1000 |
| 1 | 66.48 | 71.44 | 69.02 | 61.03 | 1730.42 | 0.0087 | 0.0665 | 0.8632 | 0 | 0 | 0 | 0 |
| | 70.21 | 68.48 | 62.82 | 66.89 | 0.0004 | 0.0143 | 0.0799 | 0.8187 | 0 | 0 | 0 | 0 |
| | 64.18 | 67.08 | 65.34 | 65.59 | 1395.40 | 0.0289 | 0.1325 | 0.8401 | 0 | 0 | 0 | 0 |
| 2 | 68.69 | 66.41 | 69.65 | 60.56 | 4690.40 | 4730.90 | 2922.10 | 2985.18 | 2751.50 | 2639.00 | 2661.70 | 2257.23 |
| | 72.66 | 69.45 | 66.50 | 62.72 | 4576.30 | 4893.40 | 4333.60 | 4456.44 | 2692.30 | 2684.80 | 2576.40 | 2264.09 |
| | 68.51 | 66.52 | 70.01 | 66.95 | 4551.40 | 4747.60 | 4486.30 | 4173.21 | 2684.10 | 2774.30 | 2610.10 | 2284.85 |
| 3 | 69.73 | 68.33 | 66.34 | 64.09 | 4593.45 | 4643.50 | 4315.30 | 4516.26 | 2741.10 | 1744.70 | 1849.90 | 2635.91 |
| | 70.11 | 67.15 | 71.44 | 62.01 | 4709.31 | 4579.20 | 4826.40 | 4638.85 | 2678.70 | 2685.10 | 2619.20 | 2658.97 |
| | 71.53 | 68.31 | 65.02 | 69.49 | 4763.50 | 4795.30 | 4766.30 | 4813.05 | 2721.90 | 2765.70 | 2680.00 | 2614.68 |
| 4 | 68.65 | 68.21 | 69.02 | 65.69 | 4622.74 | 4798.70 | 4538.10 | 4610.40 | 2770.60 | 2751.10 | 2637.80 | 2589.11 |
| | 67.34 | 68.59 | 67.36 | 66.28 | 4576.96 | 4545.30 | 4678.60 | 4506.10 | 2715.20 | 1816.60 | 1855.40 | 2692.65 |
| | 68.17 | 69.54 | 69.56 | 64.86 | 4594.81 | 4790.00 | 4614.80 | 2990.10 | 2642.20 | 2725.00 | 2651.70 | 2619.41 |

The results in Table 1 indicate that GPR algorithm can be usedto solvethe TSP,and it gives the near-optimal solution though the size of sample tours are small.In particular, the solutions of Wolverine manufacturing problem can be solved by small sample tours, with 5 tours of all possible tours ($3.63\times10^{5}$), or 0.00138% of them. Moreover, we observe that the initial hyperparameter (length-scale) and the sample size of tour affectthe quality of the algorithm when the problem is large, as shown in gr17 and fri26 problems with length-scale $\ell$ = 1. For example, GPR algorithm cannot find the solutions of fri26problem with length-scale $\ell$ = 1when its sample size is very small (e.g., 5 tours of $4.03\times10^{26}$ tours or $1.24\times10^{-24}$ % of all possible tours) while the other length-scales can be find the solutions. Therefore, the sample size of tour and the initial length-scale are necessary conditions for solving TSP by GPR algorithm.

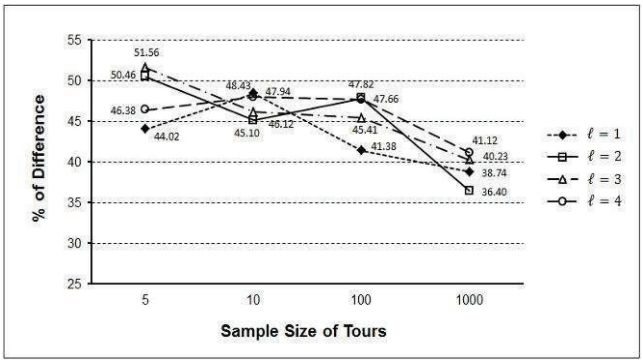


Fig. 2 The average percentage of difference between the total distance of the GPR solutions and theoptimal solution for the Wolverinemanufacturingproblem

The average percentage of difference between the total distance of the GPR solutions and the optimal solution for the Wolverine manufacturing problem has a decrease trend when the sample tours increase (as shown in Fig. 2). It is possible that the near-optimal solution can be obtained when the sample size of tour is large enough. For instance, $\ell = 2$, the average percentage of difference between the total distance of the GPR solutions and the optimal solution reduces considerably from 50.5% (with 5 sample tours) to 36.4% (with 1,000 sample tours).

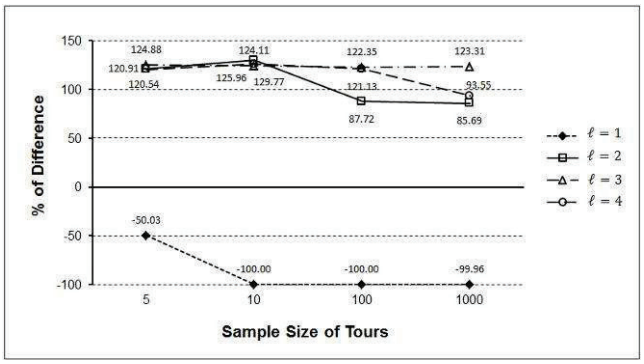


Fig. 3 The average percentage of difference between the total distance of the GPR solutions and the optimal solution for the gr17 problem

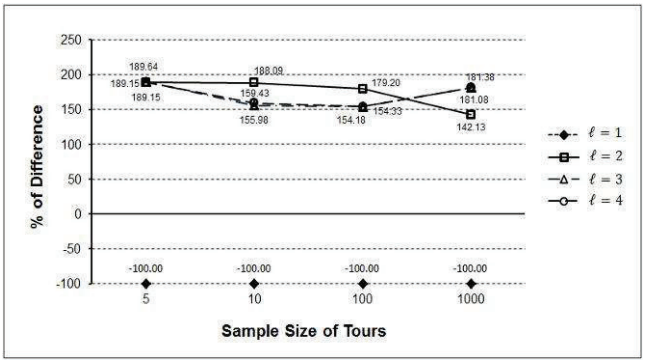


Fig. 4 The average percentage of difference between the total distance of the GPR solutions and the optimal solution for the fri26 problem

The average percentage of difference between the total distance of the GPR solutions and the optimal solution for the gr17 problem, with $\ell = 2$ and $\ell = 4$, has a slightly decrease trend when the sample tours increase while that with $\ell = 3$ has a steady trend (as shown in Fig. 3). Furthermore, in Fig. 4, the average percentage of difference between the total distance of the GPR solutions and the optimal solution for the fri26 problem with $\ell = 2$ has a moderately decrease trend when the sample tours increase while that with $\ell = 3$ and $\ell = 4$ rebound at 100 sample tours. Notice that in Fig. 3 and Fig. 4, with $\ell = 1$, the average percentage of difference between the total distance of the GPR solutions and the optimal solution is lower than zero due to the sample size of tour is too very small.

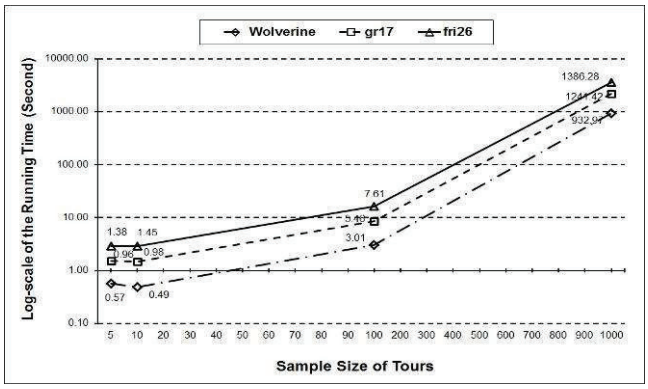


Fig. 5 The running time of the GPR algorithm among

three problems with $\ell = 2$

The computational time of all three problems are observed with four length-scales: 1, 2, 3, and 4. The experimental results indicate that the computational time for each problem is not significantly different when length-scale is changed. Therefore, we present only the computational time of the GPR algorithm for the best result (as shown in Fig. 5). The running time of the GPR algorithm among three problems with length-scales $\ell = 2$have the same trend (as shown in Fig. 5). In addition, Figure 5 show that the running time increases exponentially when the size of sample tours increase. For instance, the gr17 problemwith 5 sample toursconsumes approximately 1 second of the running time, and increase roughly to 1,000 seconds when the size of sample tours is 1,000 tours.Moreover, the size of problem also affects to the computational time, i.e., when the size of problem (the number of cities) increases, the running time increase significantly. For example, the fri26 problem with 26 cities consumes considerably more time thanthe Wolverine manufacturing problem with 9 cities.

## 5. Conclusion

We propose a method for predicting the optimal tour for a deterministic TSP with a single salesman. We formulate our problem as Gaussian process regression where the target variable is the total distance of traveling tour while the predictor is the traveling tours with the cities' number. The empirical study indicates that the size of sample tours affects to the accuracy of solution, i.e., GPR algorithm gives a solution near an optimal solution when the size of sample tours is large enough; consequently, the computational time increases.Moreover, an initial hyperparameter (length-scale) affects to solve the solution when the size of problem is large and the size of sample tours is not large enough.

However, some results of the experimentevidence that our proposed approachcan be applied to solve the TSP, andit

gives the satisfied solution for predicting the optimal total distance (when the size of sample tours is large enough) although current algorithm is not reasonable in a term of the computational time. Thus our proposed method is one of several probabilistic approaches, which is optional to solve the TSP.

For the further research, we embed the legal tour verification and tour improvement procedure into the output phase of the GPR algorithm. In addition, we generalize our proposed approach, by determiningappropriate conditions of the size of sample tours and initial length-scale, and determine a method to reduce the computational time. Furthermore, we extend this research, by considering a method for predicting the optimal total distance for the traveling salesman problem with noisy travel distance, e.g., the stochastic kriging [31, 32]. The stochastic kriging model is also considered for tour prediction.

## Acknowledgments

This conference is partially sponsored by Graduate School, Kasetsart University, Thailand. The first author would also like to thank Prince of Songkla University for their financial support of this work.

## References

[1] E. Duman, "Precedence constrained TSP arising in printed circuit board assembly," *International Journal of Production Research*, vol. 42(1), pp. 67-78, 2004.

[2] D. L. Applegate, R. E. Bixby, V. Chvátal, and W. J. Cook, The traveling salesman problem: a computational study, Princeton: University Press,New Jersey, 2006.

[3] H. Ratliff and A. Rosenthal, "Order-picking in a rectangular warehouse: a solvable case for the traveling salesman problem," in*PDRC Report Series No. 81-10*, Georgia Institute of Technology, Atlanta, Georgia., unpublished.

[4] D. Sankoff and M. Blanchette, "The median problem for breakpoints in comparative genomics," in*Proc. 3rd Annual International Conference on Computing and Combinatorics (COCOON'97)*, 20-22August,Shanghai, China, pp.251-264, 1997.

[5] P. C. Gilmore and R. E. Gomory, "Sequencing a one-state-variable machine: a solvable case of the traveling salesman problem," *Operations Research*, vol. 12, pp. 655-679, 1964.

[6] J. D. C. Little, K. G. Murty, D. W. Sweeney, and C. Karel, "An algorithm for the traveling salesman problem" *Operations Research*, vol. 11, pp. 972-989, 1963.

[7] G. Dantzig, R. Fulkerson, and S. Johnson, "Solution of a large-scale traveling-salesman problem," *Operations Research*, vol. 2, pp. 393-410, 1954.

[8] M. Bellmore and G. L. Nemhauser, "The traveling salesman problem: a survey," *Operations Research*, vol. 16, pp. 538-558, 1968.

[9] S. Lin and B. W. Kernighan, "An effective heuristic algorithm for the traveling-salesman problem," *Operations Research*, vol. 21, pp. 498-516, 1973.

[10] K. Helsgaun, "An effective implementation of the Lin- Kernighan traveling salesman heuristic," *European Journal of Operational Research*, vol. 12, pp. 106-130, 2000.

[11] K. Helsgaun, "General *k*-opt submoves for the Lin- Kernighan TSP heuristic," *Mathematical Programming Computation*, vol. 1, pp. 119-163, 2009.

[12] A. Uğur, S. Korukoğlu, A. alışkan, M. Cinsdikici, and A. Alp, "Genetic algorithm based solution for TSP on a sphere," *Mathematical and Computational Applications*, vol. 14, pp. 219-228, 2009.

[13] B. Kaur and U. Mittal, "Optimization of TSP using genetic algorithm," *Advances in Computational Sciences and Technology*, vol. 3, pp. 119-125, 2010.

[14] M. Gendreau, G. Laporte, and F. Semet, "A tabu search heuristic for the undirected selective travelling salesman problem," *European Journal of Operational Research*, vol. 106, pp. 539-545, 1998.

[15] M. Dorigo and L. M. Gambardella, "Ant colony system: a cooperative learning approach to the traveling salesman problem," *IEEE Transactions on Evolutionary Computation*, vol. 1, no. 1, pp. 53-66, 1997.

[16] X. H. Shi, Y. C. Liang, H. P. Lee, C. Lu, and Q. X. Wang, "Particle swarm optimization based algorithms for TSP and generalized TSP." *Information Processing Letters*, vol. 103, pp. 169-176.2007.

[17] X. Geng, Z. Chen, W. Yang, D. Shi, and K. Zhao, "Solving the traveling salesman problem based on an adaptive simulated annealing algorithm with greedy search,"*Applied Soft Computing*, vol.11, pp. 3680-3689, 2011.

[18] P-T. De Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein, "A Tutorial on the Cross-Entropy," *Annals of Operations Research*, vol. 134, no. 1, pp. 19-67, 2005.

[19] C. E. Rasmussen and C. Williams, Gaussian processes for machine learning, MIT Press, Massachusetts, 2006.

[20] F. Sinz, J. Quiñonero-Candela, G. H. Bakir, C. E. Rasmussen, and M. O. Franz, "Learning depth from stereo," in*Proc. 26th DAGM Symposium*,30August -1 September, Berlin, German, pp. 245-252, 2004.

[21] A. Krause, A. Singh, and C. Guestrin, "Near-optimal sensor placements in Gaussian processes: Theory, Efficient Algorithms and Empirical Studies," *Journal of Machine Learning Research*, vol. 9, pp. 235-284, 2008.

[22] J. C. Platt, C. J. C. Burges, S. Swenson, C. Weare, and A. Zheng, "Learning a Gaussian process prior for automatically generating music playlists," *Advances in Neural Information Processing Systems*, vol. 13, pp. 1425-1432, 2001.

[23] J. Ko, D. J. Klein, D. Fox, and D. Haehnel, "Gaussian processes and reinforcement learning for identification and control of an autonomous blimp," in *Proc. International Conference on Robotics and Automation (ICRA)*, 10-14 April,Rome, Italy, pp. 742-747, 2007.

[24] T. Idé and S. Kato, "Travel-time prediction using Gaussian process regression: a trajectory-based approach," in *Proc. 9th SIAM International Conference on Data Mining*, 30 April - 2 May, Nevada, USA, pp. 1185-1196, 2009.

[25] P. Hu, "RANDPERMS – a random permutation function," http://www.mathworks.com/matlabcentral/fileexchange/8276, 2005.

[26] P. Larrañaga, C.M.H. Kuijpers, R.H. Murga, I. Inza and S. Dizdarevic, "Genetic algorithms for the travelling salesman problem: a review of representations and operators."*Artificial Intelligence Review*, vol. 13, pp. 129-170, 1999.

[27] G. Laporte, “The traveling salesman problem: an overview of exact and approximate algorithms,”*European Journal of Operational Research*, vol. 59, pp. 231-247, 1992.

[28] C.T. Ragsdale, Spreadsheet modeling and decision analysis: a practical introduction to management science, 5th ed.(revised), Thomson Higher Education, Ohio, pp. 385-389, 2007.

[29] G. Reinelt, “TSPLIB - A Traveling Salesman Problem Library,” http://elib.zib.de/pub/mp-testdata/tsp/tsplib/tsplib.html., 1995.

[30] B. L. Nelson and D. Goldsman, "Comparisons with a standard in simulation experiments," *Management Science*, vol. 47, no. 3, pp. 449–463, 2001.

[31] B. Ankenman, B. L. Nelson, and J. Staum, “Stochastic kriging for simulation metamodeling,” *Operations Research*, vol. 58, no. 2, pp. 371–382, 2010.

[32] J. Yin, S. H. Ng, and K. M. Ng, “A study on the effects of parameter estimation on kriging model’s prediction error in stochastic simulations,” in *Proc. of the 2009 Winter Simulation Conference*, 13-16 December,Austin, Texas, USA, pp. 674-685, 2009.