Yokohama Publishers
ISSN 1349-8169 ONLINE JOURNAL

Pacific Journal of Optimization
Vol. 15, No. 1, pp. 91 – 110, 2019

SOLVING THE LOGIT-BASED STOCHASTIC USER EQUILIBRIUM USING MODIFIED PROJECTED CONJUGATE GRADIENT METHOD VIA CONVEX MODEL*

Chao Zhang†, Qian Zhang and Naihua Xiu

Abstract: In this paper, we convert the classic stochastic user equilibrium (SUE) into a convex model and propose the modified projected conjugate gradient (mPCG) method to solve the logit-based SUE. We show that the mPCG method converges to the unique global minimizer of the convex model. Numerical experiments on Sioux Falls network of moderate size are done to compare the mPCG method solving the proposed convex model, with the projected gradient (PG) method for the convex model and the method of the successive averages (MSA) for the nonconvex model. The numerical results indicate that the proposed mPCG method for the convex model outperforms the MSA and the PG method.

Key words: *stochastic user equilibrium; Convex model; Modified projected conjugate gradient method*

Mathematics Subject Classification: *65K05; 90B15; 90C25*

1 Introduction

The traffic equilibrium investigates the stable traffic assignment, which plays important role in transportation planning. The static user equilibrium (UE) was first proposed by Wardrop [16] in 1952, and has invoked a lot of interests since then. In equilibrium state, all the travelers are assumed to have perfect information about the travel cost of the network, and hence always choose the paths with smallest travel cost for each origin-destination pair. That is, no traveler can decrease his/her travel cost by unilaterally changing path in UE.

Wardrop's user equilibrium puts strong assumption on traveler's behavior of path choice, which assumes each traveler grasps the perfect information and makes the best decision. In reality, however, this assumption is hard to accomplish. Daganzo and Sheffi (1977) [8] extended the UE to the stochastic user equilibrium (SUE), and gave its unconstrained non-convex optimization reformulation. In the SUE, no traveler can decrease his/her perceived travel cost by unilaterally changing his/her route. The perceived travel cost allows stochastic factors, which reflects that different travelers may feel different travel costs of the same path, and therefore more suitable to describe real-world traffic system. There are two types of SUE models, based on the different types of distributions for the perceived travel cost.

*The work was supported in part by the National Natural Science Foundation of China (NSFC) under Grants 11571033, 11431002, and the Fundamental Research Funds for the Central Universities of China under Grant 2016JBZ012.

†Corresponding author.

The logit-based SUE is widely used and is attractive for large-scale network because of the advantage of computation, which assumes the perceived travel cost is a random vector followed by Gumbel distribution. The probit-based SUE assumes that the perceived travel cost follows the normal distribution, which is more appropriate to reality but more difficult to solve.

Invoked by the SUE, various extensions that reflect more realistic stochastic equilibrium models have been proposed. Maher (2001) [8] extended the classic SUE to the stochastic user equilibrium assignment with elastic demand. Watling (2002) [17] proposed a second order stochastic network equilibrium model. Based on the model [17], Clark and Watling (2005) [5] proposed a stochastic network model which assumes that the stochastic travel demand follows a stationary Poisson process and a probabilistic route choice model. Nakayama and Takayama (2003) [11] suggested a similar stochastic user equilibrium model which assumes the travel demand follows the Binormal distribution. Lam et al. (2008) [6] proposed a stochastic network model in which both the demand and supply are stochastic due to adverse weather condition. Sumalee et al. (2009) [**?**] considered network equilibrium under endogenous stochastic demand and supply. Agdeppa et al. (2010) [1] proposed a convex residual model for stochastic affine variational inequalities problems to address the traffic equilibrium under uncertainty. Sumalee et al. (2011) [15] addressed stochastic multi-modal transport network under demand uncertainties and adverse weather condition. Zhang et al. (2011) [19] considered robust Wardrop's user equilibrium under stochastic demand and supply via expected residual minimization approach.

Efficient algorithms are urged to solve the SUE and its extensions due to their widely applications. The method of successive average (MSA) was used by Sheffi and Powell (1985) [13]. This method employs the predetermined step sizes and provides a stable and effective way for solving the SUE. The MSA has been widely used, however, the convergence result is not always guaranteed as pointed out in [7]. Liu et al. (2009) [7] modified the MSA and gave the method of successive weighted average (MSWA). The MSWA modifies the predetermined step sizes and improves the computational speed and quality to some degree, compared to the MSA. Meng et al. (2009) [10] proposed a PC-CA algorithm for solving the probit-based SUE model. Yu et al. (2014) [18] considered the logit-based SUE model with elastic demand, which is eventually transformed to a classic SUE with fixed demand. The problem is then converted to a linearly convex programming and a predictor-corrector interior point algorithm is suggested to solve it.

From the view of optimization, whether the model is convex or not affects a lot the performance of the algorithms [2]. In this paper, instead of using the unconstrained/bound-constrained nonconvex optimization model for SUE, we use the bound constrained convex reformulation. We propose the modified projected conjugate gradient (mPCG) method to solve the convex programming with solid convergent result. Numerical results demonstrate the necessity of the convex reformulation, and the good performance of the mPCG method, compared to the MSA and the PG method.

The rest of this paper is organized as follows. Section 2 gives a brief description of the logit-based SUE model and its classic unconstrained nonconvex optimization reformulation, as well as the bound constrained nonconvex optimization reformulation. The nonconvex property is analyzed. Section 3 transforms the nonconvex programming into the convex programming. Section 4 provides the modified projected conjugate gradient method that addresses the SUE problem and show its convergence. Section 5 presents the numerical experiments on the Sioux-Falls network using different logit assignment parameters.

$\boxed{2}$ SUE Model Formulation

First of all, we outline the notations and make a brief description about the SUE. We also present the nonconvex unconstrained/bound constrained reformulations of SUE and analyze their nonconvex property. Let us consider a general directed traffic network $\mathcal{G} = [\mathcal{N}, \mathcal{A}]$, where $\mathcal{N}$ is the set of nodes, and $\mathcal{A}$ is the set of directed links. Let us denote by $\mathcal{W}$ the set of origin-destination (OD) pairs, and $\mathcal{K}_w$ the set of all "available" paths connecting the OD pair $w \in \mathcal{W}$. We assume that the network is strongly connected, which means that there is at least one path connecting each OD pair w. Let x_a be the flow on the link a, f_k^w be the flow on the path $k \in \mathcal{K}_w$, and D_w be the demand connecting the OD pair w, respectively. We also use $\mathbf{x} = (x_a)_{a\in\mathcal{A}}$ to denote the link flow vector and $\mathbf{f}^w = (\dots, f_k^w, \dots)^T_{k\in\mathcal{K}_w}$ to denote the path flow vector corresponding to the OD pair $w \in \mathcal{W}$. It is easy to see from the flow conservation rules that for any a, k, w,

$$x_a = \sum_{w\in\mathcal{W}} \sum_{k\in\mathcal{K}_w} \delta_{a,k} f_k^w, \tag{2.1}$$

$$\sum_{k\in\mathcal{K}_w} f_k^w = D_w, \tag{2.2}$$

where $\delta_{a,k} = 1$ if link a is on the path k, and $\delta_{a,k} = 0$ otherwise.

Let $t_a = t_a(x_a)$ be the mean travel time for each link a, i.e., $t_a = E[T_a]$ where T_a is the perceived travel time on the link a. Let C_k^w and c_k^w be the perceived travel cost and the actual travel cost of path $k \in \mathcal{K}_w$, where

$$C_k^w = \sum_a T_a \delta_{a,k} \quad \text{and} \quad c_k^w = \sum_a t_a \delta_{a,k}. \tag{2.3}$$

The difference between the stochastic user equilibrium (SUE) to the static user equilibrium (UE) lies in the introduction of path choice probability. The path choice probability on a certain path $k \in \mathcal{K}_w$ is not 0 or 1 as in the UE, but a number within the interval $[0, 1]$ which reflects the stochastic characteristic of the path choice since the travelers may not grasp the travel time precisely due to incompleteness of information. The probability p_k^w of choosing the path $k \in \mathcal{K}_w$ is defined as

$$p_k^w = \Pr\{C_k^w \le C_l^w,\ \forall l \in \mathcal{K}_w\},$$

where

$$C_k^w = c_k^w - \frac{1}{\theta}\epsilon_k^w, \tag{2.4}$$

with ϵ_k^w being a random term associated with the path under consideration and θ being a given parameter. The SUE is a fixed point problem

$$f_k^w = D_w p_k^w, \tag{2.5}$$

for all k and w. Let us denote by $\mathbf{c}^w(\mathbf{x}) = (\dots, c_k^w(\mathbf{x}), \dots)^T_{k\in\mathcal{K}_w}$ at a given flow level $\mathbf{x}$. The SUE can be obtained by solving the following unconstrained minimization problem (Page 312 of [12]))

$$\min_{\mathbf{x}} z(\mathbf{x}) := -\sum_w D_w S_w[\mathbf{c}^w(\mathbf{x})] + \sum_a x_a t_a(x_a) - \sum_a \int_0^{x_a} t_a(\omega)d\omega. \tag{2.6}$$

The expected minimum perceived travel time $S_w[\mathbf{c}^w(\mathbf{x})]$ in (2.6) is defined as

$$S_w[\mathbf{c}^w(\mathbf{x})] = E[\min_{k\in\mathcal{K}_w}\{C_k^w\} \mid \mathbf{c}^w(\mathbf{x})],$$

which is concave with respect to $\mathbf{c}^w(\mathbf{x})$, and has the partial derivative

$$\frac{\partial S_w[\mathbf{c}^w(\mathbf{x})]}{\partial c_k^w} = p_k^w. \tag{2.7}$$

By the analysis of why (2.6) leads to the SUE (Page 317 of [12]), we know that SUE can be obtained by any unconstrained minimization program with objective function $\hat{z}(\mathbf{x})$ that is modified from $z(\mathbf{x})$ by replacing $S_w[c^w(\mathbf{x})]$ in (2.6) by $\hat{S}_w[c^w(\mathbf{x})]$, as long as

$$\frac{\partial \hat{S}_w[\mathbf{c}^w(\mathbf{x})]}{\partial c_k^w} = \frac{\partial S_w[\mathbf{c}^w(\mathbf{x})]}{\partial c_k^w} = p_k^w. \tag{2.8}$$

In this paper, we assume ϵ_k^w in (2.4) is identically and independently distributed (i.i.d.) Gumbel variable. The path choice probabilities are then the logit path choice probabilities

$$p_k^w = \frac{e^{-\theta c_k^w}}{\sum_{l\in\mathcal{K}_w} e^{-\theta c_l^w}}. \tag{2.9}$$

The logit-based SUE is a fixed point problem

$$f_k^w = D_w p_k^w = D_w \frac{e^{-\theta c_k^w}}{\sum_{l\in\mathcal{K}_w} e^{-\theta c_l^w}}. \tag{2.10}$$

Assume the mean travel time is the usually used Bureau of Public Roads (BPR) function

$$t_a(x_a) = t_a^0\left[1 + \beta_a\left(\frac{x_a}{R_a}\right)^{n_a}\right], \tag{2.11}$$

where the parameters $\beta_a > 0$, and $n_a > 0$ is a positive integer, and $t_a^0 = t_a(0)$ and R_a are the free-flow travel time and the capacity of the link a, respectively. According to (2.8) and (2.9), we can choose

$$\hat{S}_w[\mathbf{c}^w(\mathbf{x})] = -\frac{1}{\theta}\ln\sum_{l\in\mathcal{K}_w} e^{-\theta c_l^w}. \tag{2.12}$$

Because by using Chain rule, we get

$$\begin{aligned}
\frac{\partial \hat{S}_w[\mathbf{c}^w(\mathbf{x})]}{\partial c_k^w} &= \frac{\partial(-\frac{1}{\theta}\ln\sum_{l\in\mathcal{K}_w} e^{-\theta c_l^w})}{\partial c_k^w} \\
&= \frac{d(-\frac{1}{\theta}\ln\sum_{l\in\mathcal{K}_w} e^{-\theta c_l^w})}{d(\sum_{l\in\mathcal{K}_w} e^{-\theta c_l^w})}\frac{\partial(\sum_{l\in\mathcal{K}_w} e^{-\theta c_l^w})}{\partial c_k^w} \\
&= -\frac{1}{\theta}\frac{1}{\sum_{l\in\mathcal{K}_w} e^{-\theta c_l^w}}\frac{\partial(\sum_{l\in\mathcal{K}_w} e^{-\theta c_l^w})}{\partial c_k^w} \qquad (2.13)\\
&= -\frac{1}{\theta}\frac{1}{\sum_{l\in\mathcal{K}_w} e^{-\theta c_l^w}}(-\theta e^{-\theta c_k^w})
\end{aligned}$$

$$= \frac{e^{-\theta c_k^w}}{\sum_{l\in\mathcal{K}_w} e^{-\theta c_l^w}},$$

which satisfies (2.8) and (2.9). And hence we define

$$\hat{z}(\mathbf{x}) := -\sum_w D_w \hat{S}_w[\mathbf{c}^w(\mathbf{x})] + \sum_a x_a t_a(x_a) - \sum_a \int_0^{x_a} t_a(\omega)d\omega \tag{2.14}$$

$$= \frac{1}{\theta}\sum_w D_w \ln \sum_{l\in\mathcal{K}_w} e^{-\theta c_l^w} + \sum_a \frac{t_a^0\beta_a}{R_a^{n_a}}(1-\frac{1}{n_a+1})x_a^{n_a+1}. \tag{2.15}$$

The SUE can be obtained by the following unconstrained programming

$$\min\ \hat{z}(\mathbf{x}), \tag{2.16}$$

in the sense that if $\mathbf{x}^* > 0$ is a stationary point of (2.16), then $\mathbf{f}^* = D_w\mathbf{P}^w(\mathbf{c}^w(\mathbf{x}^*))$ satisfies the SUE condition (2.5), and $\mathbf{x}^*$ and $\mathbf{f}^*$ are the link flow pattern and the path flow pattern that satisfy the flow conservation rules (2.1) and (2.2).

Since we seek $\mathbf{x}^*$ as link flow pattern which should be nonnegative, we may also consider the bound constrained programming for SUE

$$\begin{array}{ll} \min & \hat{z}(\mathbf{x}) \\ \text{s.t.} & \mathbf{x} \geq 0. \end{array} \tag{2.17}$$

Similarly, if $\mathbf{x}^* > 0$ is a stationary point of (2.17), then $\mathbf{f}^* = D_w\mathbf{P}^w(\mathbf{c}^w(\mathbf{x}^*))$ satisfies the SUE condition (2.5), and $\mathbf{x}^*$ and $\mathbf{f}^*$ are the link flow pattern and the path flow pattern that satisfy the flow conservation rules (2.1) and (2.2).

Let $\Delta_w = (\delta_{a,k})_{a\in\mathcal{A},k\in\mathcal{K}_w}$ be the link-path incidence matrix corresponding to the OD pair $w \in \mathcal{W}$. Let us denote the vectors $\mathbf{t} = \mathbf{t}(\mathbf{x}) = (\dots, t_a(x_a), \dots)^T_{a\in\mathcal{A}}$, $\nabla\mathbf{t} = \nabla\mathbf{t}(\mathbf{x}) = (\dots, t'_a(x_a), \dots)^T_{a\in\mathcal{A}}$ where $t'_a(x_a)$ is the first order derivative of t_a with respect to x_a, and the diagonal matrix

$$\nabla^2\mathbf{t} = \nabla^2\mathbf{t}(\mathbf{x}) = \text{diag}\left((\dots, t''_a(x_a), \dots)^T_{a\in\mathcal{A}}\right).$$

Here $t''_a(x_a)$ is the second order derivative of $t_a(x_a)$ with respect to x_a. Denote the vectors $\mathbf{P}^w = (\dots, p_k^w, \dots)^T_{k\in\mathcal{K}_w}$ and $\nabla_{\mathbf{c}}\mathbf{P}^w = (\dots, \frac{\partial p_k^w}{\partial c_k^w}, \dots)^T_{k\in\mathcal{K}_w}$ where

$$\frac{\partial p_k^w}{\partial c_k^w} = -\frac{\theta e^{-\theta c_k^w}\sum_{l\neq k, l\in\mathcal{K}_w} e^{-\theta c_l^w}}{\sum_{l\in\mathcal{K}_w} e^{-\theta c_l^w}}. \tag{2.18}$$

By direct computation, the Hessian matrix of $\hat{z}(\mathbf{x})$ in (2.14) is

$$\nabla^2\hat{z}(\mathbf{x}) = \sum_w D_w(\Delta_w^T\nabla\mathbf{t})^T\text{diag}(-\nabla_{\mathbf{c}}\mathbf{P}^w)(\Delta_w^T\nabla\mathbf{t}) + \text{diag}(\nabla\mathbf{t}) + (\nabla^2\mathbf{t})V, \tag{2.19}$$

where $V = \text{diag}(\mathbf{x} - \sum_w D_w\Delta_c\mathbf{P}^w)$. The first term in the right-hand side of (2.19) is positive semidefinite by noting (2.8) and the concavity of $S_w[\mathrm{c}^w(\mathbf{x})]$, and the second term in the right-hand side of (2.19) is positive definite. However, the matrix V might be an indefinite matrix corresponding to the flow pattern $\mathbf{x}$. This indicates that $\nabla^2\hat{z}(\mathbf{x})$ is not positive semidefinite for all $\mathbf{x}$, Thus the function $\hat{z}(\mathbf{x})$ is not convex and the programming defined in (2.16) for SUE is not a convex programming.

$\boxed{3}$ SUE Convex Programming

Although the SUE reformulation given in (2.16) is an unconstrained minimization programming with explicit formula to calculate the objective function (2.15). Its nonconvexity brings difficulty of employing efficient algorithms. If we transform the nonconvex programming (2.16) to a strictly convex programming whose unique minimizer provides the flow pattern that satisfies the SUE defined in (2.5), then efficient algorithms can be applied with solid convergent result. In this section, we focus on obtaining a convex reformulation of the SUE.

Note that the link flow x_a is nonnegative for all $a \in \mathcal{A}$. By the BPR function in (2.11), it is easy to see that the link travel time t_a and the link flow x_a has the one-to-one correspondence. Let us denote by $x_a(t_a)$ the inverse function of the BPR function $t_a(x_a)$ in (2.11). Then

$$x_a(t_a) = R_a\left(\frac{t_a - t_a^0}{\beta_a t_a^0}\right)^{\frac{1}{n_a}} \quad \text{for } t_a \geq t_a^0. \tag{3.1}$$

Let $\mathbf{t}^0 := \mathbf{t}(0) = (\ldots, t_a^0, \ldots)^T$, and

$$\hat{h}(\mathbf{t}) \quad := \quad -\sum_w D_w \hat{S}_w[\mathbf{c}^w(\mathbf{x}(\mathbf{t}))] + \sum_a \int_{t_a^0}^{t_a} x_a(\nu)d\nu \quad \text{for } \mathbf{t} \geq \mathbf{t}^0. \tag{3.2}$$

Lemma 3.1. *For any* $\mathbf{t} \geq \mathbf{t}^0$,

$$\hat{h}(\mathbf{t}) = \hat{z}(\mathbf{x}(\mathbf{t})) = \hat{z}(\mathbf{x}). \tag{3.3}$$

Proof. Letting $t_a(\omega) = \nu$ and using integration by parts, we find the last term of $\hat{z}(\mathbf{x})$ in (2.14) equals

$$\begin{aligned}
\int_0^{x_a} t_a(\omega)d\omega &= \int_{t_a(0)}^{t_a(x_a)} \nu dx_a(\nu) \\
&= \nu x_a(\nu)|_{t_a^0}^{t_a} - \int_{t_a^0}^{t_a} x_a(\nu)d\nu \\
&= t_a x_a(t_a) - t_a^0 x_a(t_a^0) - \int_{t_a^0}^{t_a} x_a(\nu)d\nu \\
&= t_a x_a(t_a) - t_a^0 \cdot 0 - \int_{t_a^0}^{t_a} x_a(\nu)d\nu \\
&= x_a(t_a)t_a - \int_{t_a^0}^{t_a} x_a(\nu)d\nu,
\end{aligned}$$

where the fourth equation is obtained from $x_a(t_a^0) = 0$ according to (3.1). Hence from (2.14),

$$\begin{aligned}
\hat{z}(\mathbf{x}) &= \hat{z}(\mathbf{x}(\mathbf{t})) \\
&= -\sum_w D_w \hat{S}_w[\mathbf{c}^w(\mathbf{x}(\mathbf{t}))] + \sum_a x_a t_a(x_a) - \sum_a \int_0^{x_a} t_a(\omega)d\omega \\
&= -\sum_w D_w \hat{S}_w[\mathbf{c}^w(\mathbf{x}(\mathbf{t}))] + \sum_a x_a t_a(x_a) - \sum_a x_a(t_a)t_a + \sum_a \int_{t_a^0}^{t_a} x_a(\nu)d\nu
\end{aligned}$$

$$
\begin{aligned}
&= -\sum_w D_w \hat{S}_w[\mathbf{c}^w(\mathbf{x}(\mathbf{t}))] + \sum_a \int_{t_a^0}^{t_a} x_a(\nu) d\nu \\
&= \hat{h}(\mathbf{t}).
\end{aligned}
$$

□

Lemma 3.2. *For* $\mathbf{t} \geq \mathbf{t}^0$, *if* $\|\mathbf{t}\| \to +\infty$, *then* $\hat{h}(\mathbf{t}) \to +\infty$.

Proof. Using (3.1), we find

$$
\begin{aligned}
\int_{t_a^0}^{t_a} x_a(\nu) d\nu &= \int_{t_a^0}^{t_a} R_a \left(\frac{\nu - t_a^0}{\beta_a t_a^0} \right)^{\frac{1}{n_a}} d\nu \\
&= \int_{t_a^0}^{t_a} R_a \beta_a t_a^0 \left(\frac{\nu - t_a^0}{\beta_a t_a^0} \right)^{\frac{1}{n_a}} d\left(\frac{\nu - t_a^0}{\beta_a t_a^0} \right).
\end{aligned}
$$

Let $\frac{\nu - t_a^0}{\beta_a t_a^0} = \delta$. Then $\nu \in [t_a^0, t_a]$ indicates that $\delta \in [0, \frac{t_a - t_a^0}{\beta_a t_a^0}]$. Hence

$$
\begin{aligned}
\int_{t_a^0}^{t_a} x_a(\nu) d\nu &= \int_0^{\frac{t_a - t_a^0}{\beta_a t_a^0}} R_a \beta_a t_a^0 \delta^{\frac{1}{n_a}} d\delta \\
&= R_a \beta_a t_a^0 \left. \frac{\delta^{\frac{1}{n_a}+1}}{\frac{1}{n_a}+1} \right|_0^{\frac{t_a - t_a^0}{\beta_a t_a^0}} \\
&= \frac{n_a}{n_a+1} R_a \beta_a t_a^0 \left(\frac{t_a - t_a^0}{\beta_a t_a^0} \right)^{\frac{1}{n_a}+1}. \qquad (3.4)
\end{aligned}
$$

Substituting (2.12) and (3.4) into (3.2), we get for all $\mathbf{t} \geq \mathbf{t}^0$

$$
\hat{h}(\mathbf{t}) = \frac{1}{\theta} \sum_w D_w \ln \sum_{l \in \mathcal{K}_w} e^{-\theta c_l^w} + \sum_a \frac{n_a}{n_a+1} R_a \beta_a t_a^0 \left(\frac{t_a - t_a^0}{\beta_a t_a^0} \right)^{\frac{1}{n_a}+1}. \qquad (3.5)
$$

By (2.3),

$$
c_l^w = \sum_a t_a \delta_{a,l}
$$

where $\delta_{a,l} = 1$ if link a is on the path l and $\delta_{a,l} = 0$ otherwise. It is easy to see that

$$
e^{-\theta c_l^w} = e^{-\theta(\sum_a t_a \delta_{a,l})} \geq e^{-\theta(\sum_a t_a)},
$$

and consequently

$$
\begin{aligned}
\ln \sum_{l \in \mathcal{K}_w} e^{-\theta c_l^w} &\geq \ln \sum_{l \in \mathcal{K}_w} e^{-\theta(\sum\limits_a t_a)} \\
&= \ln |\mathcal{K}_w| e^{-\theta(\sum\limits_a t_a)} \\
&= \ln |\mathcal{K}_w| + \ln e^{-\theta(\sum\limits_a t_a)} \\
&= \ln |\mathcal{K}_w| - \theta \sum_a t_a.
\end{aligned}
$$

Letting $\bar{D} := \sum\limits_w D_w$, and for each $a \in \mathcal{A}$,

$$\varrho_a := \frac{n_a}{n_a+1} R_a (\beta_a t_a^0)^{-\frac{1}{n_a}},$$

which is a positive constant, we find

$$\begin{aligned}
\hat{h}(\mathbf{t}) &\geq \frac{1}{\theta}\sum_w D_w(\ln|\mathcal{K}_w| - \theta\sum_a t_a) + \sum_a \frac{n_a}{n_a+1} R_a\beta_a t_a^0 \left(\frac{t_a - t_a^0}{\beta_a t_a^0}\right)^{\frac{1}{n_a}+1} \\
&= \frac{1}{\theta}\sum_w D_w \ln|\mathcal{K}_w| - \sum_w D_w \sum_a t_a + \sum_a \frac{n_a}{n_a+1} R_a\beta_a t_a^0 \left(\frac{t_a - t_a^0}{\beta_a t_a^0}\right)^{\frac{1}{n_a}+1} \\
&= \frac{1}{\theta}\sum_w D_w \ln|\mathcal{K}_w| - \bar{D}\sum_a t_a + \sum_a \frac{n_a}{n_a+1} R_a\beta_a t_a^0 \left(\frac{t_a - t_a^0}{\beta_a t_a^0}\right)^{\frac{1}{n_a}+1} \\
&= \frac{1}{\theta}\sum_w D_w \ln|\mathcal{K}_w| + \sum_a \left[\varrho_a (t_a - t_a^0)^{\frac{1}{n_a}+1} - \bar{D} t_a\right]. \qquad (3.6)
\end{aligned}$$

Note that $n_a > 0$. For any $a \in \mathcal{A}$ such that $t_a \to +\infty$, we can easily find that

$$\lim_{t_a \to +\infty} \frac{\varrho_a (t_a - t_a^0)^{\frac{1}{n_a}+1} - \bar{D} t_a}{t_a} = \lim_{t_a \to +\infty} \left[\varrho_a (1 - \frac{t_a^0}{t_a})(t_a - t_a^0)^{\frac{1}{n_a}} - \bar{D}\right] = +\infty,$$

which indicates

$$\lim_{t_a \to +\infty} \left[\varrho_a (t_a - t_a^0)^{\frac{1}{n_a}+1} - \bar{D} t_a\right] = +\infty. \qquad (3.7)$$

Combining (3.7) and (3.6), we know that for any $\mathbf{t} \geq \mathbf{t}^0$, if $\|\mathbf{t}\| \to +\infty$, then

$$\hat{h}(\mathbf{t}) \to +\infty$$

as we desired. □

Now we consider the bound constrained minimization problem

$$\begin{array}{ll} \min & \hat{h}(\mathbf{t}) \\ \text{s.t.} & \mathbf{t} \geq \mathbf{t}^0. \end{array} \qquad (3.8)$$

This bound constrained minimization problem has nice convex property and can be considered a reformulation of the SUE as shown in the following theorem.

Theorem 3.3. *The bound constrained minimization problem (3.8) is a strictly convex programming that has a unique minimizer* $\mathbf{t}^* \geq \mathbf{t}^0$. *Moreover,*

(i) $\mathbf{x}^* := \mathbf{x}(\mathbf{t}^*) \geq 0$ *is a stationary point of (2.16), and* $\mathbf{x}^*$ *is the unique global minimizer of the bound constrained nonconvex reformulation of SUE in (2.17).*

(ii) $(\mathbf{f}^w)^* = D_w \mathbf{P}^w(\mathbf{c}^w(\mathbf{x}^*))$ *satisfies the SUE in (2.5), and* $\mathbf{x}^*$ *and* $(\mathbf{f}^w)^*$ *are the link flow pattern and the path flow pattern that satisfy the flow conservation rules in (2.1) and (2.2).*

Proof. For any $\mathbf{t} \geq \mathbf{t}^0$, we have by direct computation

$$\nabla \hat{h}(\mathbf{t}) = -\sum_w D_w \Delta_w \mathbf{P}^w + \mathbf{x}(\mathbf{t}) \tag{3.9}$$

and the Hessian matrix

$$\nabla^2 \hat{h}(\mathbf{t}) = \sum_w D_w \Delta_w \mathrm{diag}(-\nabla_{\mathbf{c}} \mathbf{P}^w) \Delta_w^T + \mathrm{diag}(\nabla_{\mathbf{t}} \mathbf{x}(\mathbf{t})). \tag{3.10}$$

Clearly

$$D_w \Delta_w (-\nabla_{\mathbf{c}} \mathbf{P}^w) \Delta_w^T \quad \text{is positive semidefinite}$$

for all $w \in \mathcal{W}$. From $x_a(t_a)$ in (3.1), we have

$$x_a'(t_a) = R_a \left(\frac{t_a - t_a^0}{\beta_a t_a^0} \right)^{\frac{1}{n_a} - 1} \frac{1}{\beta_a t_a^0}.$$

It is clear that $x_a'(t_a)$ is positive for all $t_a \geq t_a^0$ if $n_a = 1$. If $n_a > 1$, $x_a'(t_a) > 0$ for all $t_a > t_a^0$ and

$$x_a'(t_a^0) := \lim_{t_a \downarrow t_a^0} x_a'(t_a) = +\infty.$$

Hence we know that

$$\mathrm{diag}(\nabla_{\mathbf{t}} \mathbf{x}(\mathbf{t})) \quad \text{is positive definite for } \mathbf{t} \geq \mathbf{t}^0.$$

Therefore, the Hessian matrix $\nabla^2 \hat{h}(\mathbf{t})$ is positive definite for $\mathbf{t} \geq \mathbf{t}^0$, and the bound constrained programming in (3.8) is a strictly convex programming. We know that a strictly convex programming has at most one minimizer. Now we show that the minimizer of (3.8) exists. Let

$$\mathbf{t}^* = \inf_{\mathbf{t} \geq \mathbf{t}^0} \hat{h}(\mathbf{t}).$$

Then by Lemma 3.2, there exists an upper bound $\bar{\mathbf{t}}$ such that $\mathbf{t}^* \leq \bar{\mathbf{t}}$. Hence

$$\mathbf{t}^* = \inf_{\mathbf{t}^0 \leq \mathbf{t} \leq \bar{\mathbf{t}}} \hat{h}(\mathbf{t}).$$

Since $\hat{h}(\mathbf{t})$ is a continuous function on the compact set $\{\mathbf{t} \mid \mathbf{t}^0 \leq \mathbf{t} \leq \mathbf{t}\}$, $\hat{h}(\mathbf{t})$ can achieve its minimum $\mathbf{t}^*$ on this compact set. Hence

$$\mathbf{t}^* = \min_{\mathbf{t} \geq \mathbf{t}^0} \hat{h}(\mathbf{t}).$$

Therefore, the convex programming in (3.8) has a unique global minimizer $\mathbf{t}^*$.

Now we show the statement (i) holds. It is clear that $\mathbf{x}^* := \mathbf{x}(\mathbf{t}^*) \geq 0$ by (3.1) and $\mathbf{t} \geq \mathbf{t}^0$. By KKT optimality condition, $\mathbf{t}^*$ is the unique minimizer if and only if

$$\min\{\nabla \hat{h}(\mathbf{t}^*), \mathbf{t}^* - \mathbf{t}^0\} = 0. \tag{3.11}$$

If $t_a^* - t_a^0 > 0$, we find by (3.11) that $(\nabla \hat{h}(\mathbf{t}^*))_a = 0$, i.e.,

$$(\nabla \hat{h}(\mathbf{t}^*))_a = (-\sum_w D_w \Delta_w \mathbf{P}^w (\mathbf{c}^w(\mathbf{x}^*)))_a + \mathbf{x}_a^* = 0.$$

Otherwise $t_a^* - t_a^0 = 0$. By (3.1), we know that $x_a^* = 0$. This, combined with (3.11), yields

$$\begin{aligned}(\nabla \hat{h}(\mathbf{t}^*))_a &= (-\sum_w D_w \Delta_w \mathbf{P}^w(\mathbf{c}^w(\mathbf{x}^*)))_a + x_a^* \\ &= (-\sum_w D_w \Delta_w \mathbf{P}^w(\mathbf{c}^w(\mathbf{x}^*)))_a \\ &\geq 0.\end{aligned}$$

On the other hand,

$$(-\sum_w D_w \Delta_w \mathbf{P}^w(\mathbf{c}^w(\mathbf{x}^*)))_a \leq 0,$$

because the nonnegativity of D_w, Δ_w, and $\mathbf{P}^w(\mathbf{c}^w(\mathbf{x}^*))$. We deduce that $(\nabla \hat{h}(\mathbf{t}^*))_a = 0$. Therefore, in any case $(\nabla \hat{h}(\mathbf{t}^*))_a = 0$ and we get

$$-\sum_w D_w \Delta_w \mathbf{P}^w(\mathbf{c}^w(\mathbf{x}^*)) + \mathbf{x}^* = 0. \tag{3.12}$$

By direct computation,

$$\nabla \hat{z}(\mathbf{x}) = \left(-\sum_w D_w \Delta_w \mathbf{P}^w(\mathbf{c}^w(\mathbf{x})) + \mathbf{x}\right) .* \nabla \mathbf{t}(\mathbf{x}), \tag{3.13}$$

where ".$*$" refers to the Hadamard product that performs the product entrywise. It is clear that (3.12) implies $\nabla \hat{z}(\mathbf{x}^*) = 0$ and hence $\mathbf{x}^*$ is a stationary point of (2.16). Now we show that $\mathbf{x}^*$ is the unique global minimizer of (2.17). Suppose on the contrary that there exists $\tilde{\mathbf{x}} \geq 0$, $\tilde{\mathbf{x}} \neq \mathbf{x}^*$ such that $\hat{z}(\tilde{\mathbf{x}}) \leq \hat{z}(\mathbf{x}^*)$. Then by (2.11), there exists $\tilde{\mathbf{t}} \geq \mathbf{t}^0$ and $\tilde{\mathbf{t}} \neq \mathbf{t}^*$ such that $\hat{z}(\mathbf{x}(\tilde{\mathbf{t}})) = \hat{z}(\tilde{\mathbf{x}})$. By Lemma 3.1,

$$\hat{h}(\tilde{\mathbf{t}}) = \hat{z}(\mathbf{x}(\tilde{\mathbf{t}})) = \hat{z}(\tilde{\mathbf{x}}) \leq \hat{z}(\mathbf{x}(\mathbf{t}^*)) = \hat{h}(\mathbf{t}^*). \tag{3.14}$$

This contradicts that $\mathbf{t}^*$ is the unique global minimizer of (3.8). Hence $\mathbf{x}^*$ is the unique global minimizer of (2.17).

Now we show the statement (ii) holds. According to (3.12), we know that

$$\mathbf{x}^* = \sum_w \Delta_w D_w \mathbf{P}^w(\mathbf{c}^w(\mathbf{x}^*)) \quad \text{for all} \quad w \in \mathcal{W}. \tag{3.15}$$

Let us denote

$$(\mathbf{P}^w)^* = \mathbf{P}^w(\mathbf{c}^w(\mathbf{x}^*)) = (\dots, (p_k^w)^*, \dots)_{k \in \mathcal{K}_w}^T, \tag{3.16}$$

and set

$$(\mathbf{f}^w)^* = D_w (\mathbf{P}^w)^* \quad \text{for all} \quad w \in \mathcal{W}. \tag{3.17}$$

That is, $(f_k^w)^* = D_w (p_k^w)^*$ for all k and w, which just coincides to the SUE in (2.5). Combining (3.15), (3.16) and (3.17), we find

$$\mathbf{x}^* = \sum_w \Delta_w (\mathbf{f}^w)^*.$$

Hence the flow conservation rule in (2.1) holds. Because $(\mathbf{P}^w)^*$ is a vector of path choice probability which satisfies $\sum_{k\in\mathcal{K}_w}(p_k^w)^* = 1$, we get from (3.17) that

$$\sum_{k\in\mathcal{K}_w}(f_k^w)^* = D_w\sum_{k\in\mathcal{K}_w}(p_k^w)^* = D_w,$$

and hence (2.2) holds. □

Thus, in this paper we propose the bound constrained minimization problem in (3.8) as the convex reformulation of the SUE, in the sense that it is convex and it provides the link and the path flow pattern that satisfy the flow conservation rules (2.1) and (2.2), as well as the SUE defined in (2.5) as shown in Theorem 3.3.

Conversely, we also have the following results.

Theorem 3.4. *Let $\mathbf{x}^* > 0$ be a stationary point of the unconstrained nonconvex reformulation of SUE in (2.16). Then $\mathbf{t}^* = \mathbf{t}(\mathbf{x}^*) > \mathbf{t}^0$ is the unique global minimizer of the strictly convex programming for SUE in (3.8). And $\mathbf{x}^*$ is the unique global minimizer of the bound constrained nonconvex reformulation of SUE in (2.17).*

Proof. By (2.11), we easily find that $\mathbf{t}^* = \mathbf{t}(\mathbf{x}^*) > \mathbf{t}^0$. Since $\mathbf{x}^*$ is a stationary point of (2.16), we know that

$$\nabla\hat{z}(\mathbf{x}^*) = 0.$$

Using the formulation for $\nabla\hat{z}(\mathbf{x}^*)$ in (3.13) and the fact that $\nabla\mathbf{t}(\mathbf{x}^*) > 0$ for $\mathbf{x}^* > 0$, we find

$$\nabla\hat{h}(\mathbf{t}^*) = 0$$

according to (3.9). Hence $\mathbf{t}^*$ satisfies $\min\{\nabla\hat{h}(\mathbf{t}^*), \mathbf{t}^*\} = 0$ which indicates that $\mathbf{t}^*$ is the unique global minimizer of (3.8). By Theorem 3.3 (i), $\mathbf{x}^*$ is also the unique global minimizer of the bound constrained nonconvex reformulation of SUE in (2.17). □

Remark 3.5. In real traffic network, in general all the links will be used in the SUE. That is, $\mathbf{x}^* > 0$ or equivalently $\mathbf{t}^* > \mathbf{t}^0$ is a mild assumption. If a link is not used in the SUE, we may delete it and this will not affect the SUE for the traffic network. Let $\mathbf{t}^*$ be the unique minimizer of (3.8) and $\mathbf{x}^*$ be a stationary point of (2.16) that satisfy $\mathbf{t}^* > \mathbf{t}^0$ and $\mathbf{x}^* > 0$, respectively. Then Lemma 3.1, Theorem 3.3 and Theorem 3.4 guarantee that $\hat{h}(\mathbf{t}^*) = \hat{z}(\mathbf{x}^*)$.

4 Modified Projected Conjugate Gradient Method

We propose a modified projected conjugate gradient (mPCG) method to solve the bound constrained convex SUE programming defined in (3.8), which extends the conjugate gradient method proposed by Chen and Zhou [4] for unconstrained optimization problems.

The modified projected conjugate gradient method only uses the gradient information that is easy to implement, and is suitable for large-scale traffic network. As pointed out in [7], the usually used MSA proposed in [13] performs well in general, but the convergence to a global minimizer is not guaranteed. The predetermined stepsize may also be inefficient in computation. The modified projected conjugate gradient method here is guaranteed to converge to the global minimizer of (3.8) and uses the Armijo line search for determining the appropriate stepsize.

At follows, we always assume that the traffic network, and the parameters t_a^0, β_a, R_a, $a \in \mathcal{A}$ for the BPR function in (2.11) are given. Let us denote $\hat{h}^{(n)} = \hat{h}(\mathbf{t}^{(n)})$ and $\nabla\hat{h}^{(n)} =$

$\nabla\hat{h}(\mathbf{t}^{(n)})$, $n = 1, 2, \ldots$, for simplicity. The conjugate gradient direction $d^{(n)}$ at the nth iterate is defined as follows; see [4] for reference.

$$d^{(n)} = \begin{cases} -\nabla\hat{h}^{(n)} & \text{if } n = 1, \\ -\nabla\hat{h}^{(n)} + \zeta_n d^{(n-1)} + \tau_n u^{(n-1)} & \text{if } n \geq 1, \end{cases} \tag{4.1}$$

where

$$\zeta_n = \frac{\nabla\hat{h}^{(n)^T} u^{(n-1)}}{d^{(n-1)^T} u^{(n-1)}} - \frac{2\|u^{(n-1)}\|^2 \nabla\hat{h}^{(n)^T} d^{(n-1)}}{(d^{(n-1)^T} u^{(n-1)})^2}, \tag{4.2}$$

$$\tau_n = \frac{\nabla\hat{h}^{(n)^T} d^{(n-1)}}{d^{(n-1)^T} u^{(n-1)}}, \tag{4.3}$$

$$u^{(n-1)} = y^{(n-1)} + \eta_n s^{(n-1)}, \tag{4.4}$$

$$\eta_n = \max\{0, -\frac{s^{(n-1)^T} y^{(n-1)}}{\|s^{(n-1)}\|^2}\}, \tag{4.5}$$

with

$$y^{(n-1)} = \nabla\hat{h}^{(n)} - \nabla\hat{h}^{(n-1)}, \quad s^{(n-1)} = \mathbf{t}^{(n)} - \mathbf{t}^{(n-1)}. \tag{4.6}$$

Now we give the modified projected conjugate gradient method for solving the bound constrained convex programming in (3.8) for SUE.

At follows, we always assume that the traffic network, and the parameters t_a^0, β_a, R_a, n_a, $a \in \mathcal{A}$ for the BPR function in (2.11) are given. Given an initial link flow pattern $x_a^{(1)}$, $a \in \mathcal{A}$, we compute $t_a^{(1)} = t_a(x_a^{(1)})$ for all $a \in \mathcal{A}$ by (2.11) and get the initial point $\mathbf{t}^{(1)}$.

For $n \geq 1$, in order to compute

$$\nabla\hat{h}^{(n)} = -\sum_w D_w \Delta_w (\mathbf{P}^w)^{(n)} + \mathbf{x}^{(n)}, \tag{4.7}$$

we perform the following procedure.

We compute the actual path travel time $(c_k^w)^{(n)}$ for all $k \in \mathcal{K}_w$, $w \in \mathcal{W}$ by (2.3).

We then compute the path choice probability $(p_k^w)^{(n)}$ by (2.9) and the partial derivative $\left(\frac{\partial p_k^w}{\partial c_k^w}\right)^{(n)}$ for all $k \in \mathcal{K}_w$, $w \in \mathcal{W}$ by (2.18).

Algorithm 4.1. The modified projected conjugate gradient (mPCG) algorithm for solving (3.8)

Data: Given an initial link travel time pattern $\mathbf{t}^{(1)}$. Given the parameters $\rho, \sigma \in (0, 1)$, a positive integer $i_{\max} > 0$, and a small tolerance $\epsilon \geq 0$. Set $n = 1$. For $n \geq 1$,

Step 1. Compute the gradient of $\hat{h}$ at $\mathbf{t}^{(n)}$ by (4.7). If $\|\nabla\hat{h}(\mathbf{t}^{(n)})\| \leq \epsilon$, then stop.

Step 2. Compute the search direction $d^{(n)}$ by (4.1)-(4.6).

Step 3. If $n = 1$, determine the stepsize α_n by the Armijo line search, i.e.,

$$\alpha_n = \max\{\rho^0, \rho^1, \ldots\} \tag{4.8}$$

satisfying the Armijo line search,

$$\hat{h}(\max\{\mathbf{t}^{(n)} + \rho^i d^{(n)}, \mathbf{t}^0\}) \leq \hat{h}^{(n)} + \sigma \nabla \hat{h}^{(n)^T} \left(\max\{\mathbf{t}^{(n)} + \rho^i d^{(n)}, \mathbf{t}^0\} - \mathbf{t}^{(n)} \right). \tag{4.9}$$

Step 4. If $n > 1$, determine the stepsize α_n by the modified Armijo line search, i.e.,

$$\alpha_n = \max\{\rho^0, \rho^1, \ldots, \rho^{i_{\max}}\} \tag{4.10}$$

satisfying (4.9). If it fails to meet (4.9), we then set $d^{(n)} = -\nabla h(\mathbf{t}^{(n)})$ and use the Armijo line search (4.8) and (4.9) to find the stepsize α_n.

Step 5. Set $\mathbf{t}^{(n+1)} = \max\{\mathbf{t}^{(n)} + \alpha_n d^{(n)}, \mathbf{t}^0\}$ and $n := n + 1$.

Step 6. If $AC(\mathbf{t}^{(n)}) := \{a \in \mathcal{A} \mid t_a^{(n)} = t_a^0\} \neq \emptyset$, set $d^{(n)} = -\nabla h(\mathbf{t}^{(n)})$ and use the Armijo line search (4.8) and (4.9) to find the stepsize α_n. Set $\mathbf{t}^{(n+1)} = \max\{\mathbf{t}^{(n)} + \alpha_n d^{(n)}, \mathbf{t}^0\}$ and $n := n + 1$. Else return to Step 1.

Step 4 first implements the projected conjugate gradient step which only searches the stepsize within $i_{\max}$ trials. If it fails, then maybe the projected conjugate gradient direction is not a good search direction at the current point, and the mPCG method implements the projected gradient step, which is sure to success.

We have the following nice convergent result for Algorithm 4.1 that solves (3.8).

Theorem 4.2. *The sequence $\{\mathbf{t}^{(n)}\}$ generated by Algorithm 4.1 with $\epsilon = 0$ converges to the unique global minimizer of (3.8).*

Proof. The iterate point of the mPCG method in fact obtained from two kinds of steps:

(i) the projected conjugate gradient (PCG) step in Step 4 if it succeeds. In this case, the stepsize is determined within $i_{\max}$ trials.

(ii) the projected gradient (PG) step in Step 3, Step 4 if the PCG step fails to find the stepsize within $i_{\max}$ trials, or in Step 6.

If the PCG step is succeeded in Step 4, then by the strict convexity of $\hat{h}(\mathbf{t})$, we know that

$$\hat{h}\left(\mathbf{t}^{(n+1)}\right) \geq \hat{h}(\mathbf{t}^{(n)}) + \nabla \hat{h}^{(n)^T} \left(\mathbf{t}^{(n+1)} - \mathbf{t}^{(n)}\right).$$

This, combined with (4.9) and the fact that $\sigma \in (0, 1)$, indicates that

$$\nabla \hat{h}^{(n)^T} \left(\mathbf{t}^{(n+1)} - \mathbf{t}^{(n)}\right) \leq 0.$$

Thus $\hat{h}(\mathbf{t}^{(n+1)}) \leq \hat{h}(\mathbf{t}^{(n)})$ if $\mathbf{t}^{(n+1)}$ is obtained from $\mathbf{t}^{(n)}$ by the PCG step. If the PG step is employed from $\mathbf{t}^{(n)}$ to get $\mathbf{t}^{(n+1)}$, then it is clear that $\hat{h}(\mathbf{t}^{(n+1)}) < \hat{h}(\mathbf{t}^{(n)})$. Therefore, the sequence $\{\hat{h}(\mathbf{t}^{(n)})\}$ is nonincreasing. According to Theorem 3.3, $\hat{h}(\mathbf{t})$ has a unique minimizer $\mathbf{t}^*$ on the feasible region $\{\mathbf{t} \mid \mathbf{t} \geq \mathbf{t}^{(0)}\}$. Hence $\hat{h}(\mathbf{t}^{(n)}) \geq \hat{h}(\mathbf{t}^*)$ for all $\mathbf{t} \geq \mathbf{t}^*$ and consequently $\{\hat{h}(\mathbf{t}^{(n)})\}$ converges.

If there exists an infinite subsequence $\{n_j\} \subseteq \{1, 2, \ldots\}$ such that the PG step is invoked at $\mathbf{t}^{(n_j)}$ to get $\mathbf{t}^{n_j+1}$. By Lemma 3.2,

$$L_{\hat{h}}(\hat{h}(\mathbf{t}^{(1)})) = \{\mathbf{t} \geq \mathbf{t}^0 \mid \hat{h}(\mathbf{t}) \leq \hat{h}(\mathbf{t}^{(1)})\}$$

is nonempty and bounded. Thus the sequence $\{\mathbf{t}^{(n)}\}$ generated by Algorithm 4.1 is bounded. Following Theorem 2.3 and Theorem 2.4 of [3], we can easily show that any accumulation point $\tilde{\mathbf{t}}$ of $\mathbf{t}^{(n_j)}$ is a stationary point of (3.8). Hence $\tilde{\mathbf{t}}$ coincides to the unique global minimizer $\mathbf{t}^*$ by noting that (3.8) is a convex programming. And we also know $\{\hat{h}(\mathbf{t}^{(n)})\} \to \hat{h}(\mathbf{t}^*)$. By the strict convexity of $\hat{h}$, we deduce $\{\mathbf{t}^{(n)}\} \to \mathbf{t}^*$.

Otherwise, there exists a positive integer $\breve{n}$ such that for all $n \geq \breve{n}$, Algorithm 4.1 always employs the PCG step in Step 4. This means $AC(\mathbf{t}^{(n)}) = \{a \in \mathcal{A} \mid t_a^{(n)} = t_a^0\} = \emptyset$ for all $n \geq \breve{n}$. Hence for $n \geq \breve{n}$, the PCG step in fact becomes the CG step in [4] for unconstrained programming. To be specific, we have $\mathbf{t}^{(n+1)} = \mathbf{t}^{(n)} + \alpha_n d^{(n)}$ and

$$\hat{h}(\mathbf{t}^{(n+1)}) \leq \hat{h}^{(n)} + \sigma \alpha_n \nabla \hat{h}^{(n)^T} d^{(n)} \tag{4.11}$$

for $n \geq \breve{n}$. By Lemma 2.2 of [4],

$$\nabla \hat{h}^{(n)^T} d^{(n)} \leq -\frac{1}{2} \|\nabla \hat{h}^{(n)}\|^2.$$

Therefore, the Armijo line search guarantees

$$\hat{h}(\mathbf{t}^{(n+1)}) \leq \hat{h}(\mathbf{t}^{(n)}) - \frac{1}{2} \sigma \alpha_n \|\nabla \hat{h}^{(n)}\|^2.$$

Since $\{\hat{h}(\mathbf{t}^{(n)})\}$ converges, we find from the above inequality that $\|\nabla \hat{h}^{(n)}\| \to 0$ as $n \to \infty$. This also indicates any accumulation point of $\{\mathbf{t}^{(n)}\}$ is a stationary point of (3.8) and coincides to the unique global minimizer $\mathbf{t}^*$. Hence we also get $\{\mathbf{t}^{(n)}\} \to \mathbf{t}^*$ as we desired. □

Remark 4.3. After we obtain the unique global solution $\mathbf{t}^*$ of (3.8) using Algorithm 4.1, we easily obtain the link flow pattern by (3.15) and path flow pattern by (3.17) at SUE. It is worth mentioning that if $\mathbf{t}^* > 0$, then Step 6 can not occur infinitely many times. Because otherwise, there exists $\bar{n}$ such that for $n \geq \bar{n}$,

$$\hat{h}(\mathbf{t}^*) < \hat{h}(\mathbf{t}^{(n)}) < \min_{\mathbf{t} \geq \mathbf{t}^0,\ AC(\mathbf{t}^{(n)}) \neq \emptyset} \hat{h}(\mathbf{t}).$$

Thus $AC(\mathbf{t}^{(n)}) = \emptyset$ for $n \geq \bar{n}$, which contradicts that Step 6 occur infinitely many times. Then the PCG step in Step 4 becomes the unconstrained CG step, which is sure to succeed if $i_{\max}$ is sufficiently large.

5 Numerical Results

In this section, we do numerical experiments on the Sioux-Falls network shown in Figure 1. The network is of moderate size, which has 528 OD pairs, 76 arcs, and 1179 paths. The parameters in the BPR function in (2.11) are set to be $\beta_a = 0.15$, and $n_a = 2$ for all $a \in \mathcal{A}$.

We try three different logit assignment parameters

$$\theta = 0.1,\ 1,\ \text{and } 10$$

in (2.4), respectively. Note that a small value of θ indicates a large perception variance, with travelers using many paths, including some that may be significantly longer than the true shortest path (in terms of measured travel time). When $\theta \to 0$, it is known that the share of flow on all paths will be equal, regardless of path travel times. And when θ is large enough, the SUE will approach to the UE. We also try two different flow patterns as the initial flow patterns, respectively.

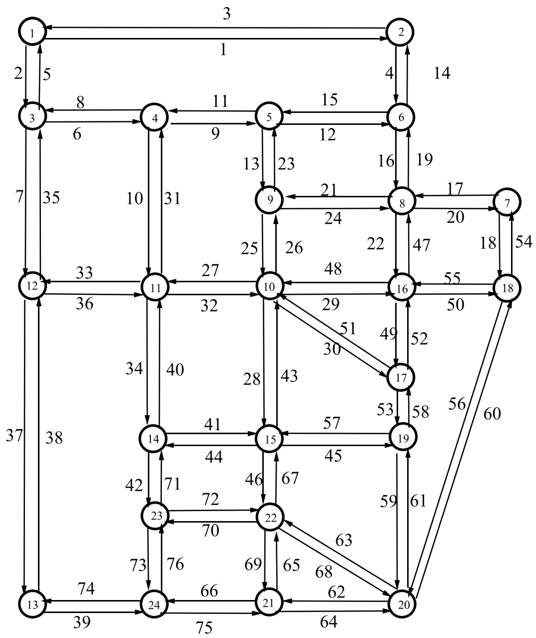


Figure 1: *Sioux Falls Network*

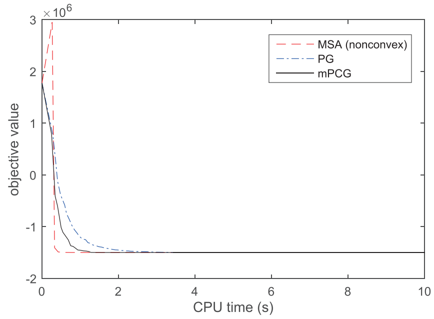
- Initial point one : it is obtained by assigning each OD demand on a single path connecting the OD pair.

- Initial point two: it is constructed by assigning each OD demand equally to all the paths connecting the OD pair.

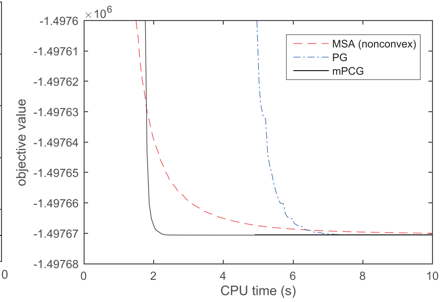
We use Matlab R2015b to implement the algorithms on a notebook with 2.50GHZ CPU and 4GB RAM. The numerical experiments show the nice performance of the mPCG method using the convex reformulation for SUE in (3.8).

Figure 2 compare the MSA for the nonconvex model, and the PG and the mPCG methods for the convex model from the first initial point using different parameters θ. And Figure 3 does the same comparison as in Figure 2 from the second initial point. Because the decrease of the objective value is of the scale 10^6 in the original figure in the left column, we select a smaller range of the objective value to see clearly the difference of the three algorithms in the right columns of Figure 2 and Figure 3.

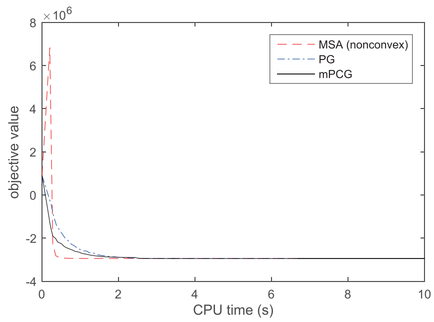
It is easy to see from the right columns of Figure 2 and Figure 3 that the mPCG method performs the best in terms of the objective value after no more than 4 seconds. Hence the convex SUE programming in (3.8) helps to implement the modified projected conjugate gradient method efficiently which has solid convergent result.



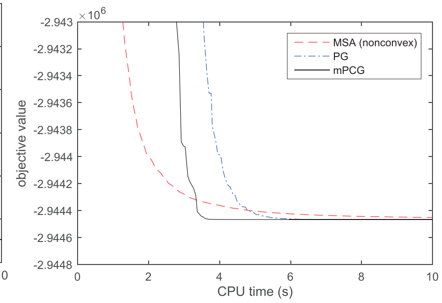
(a) $\theta = 0.1$, original



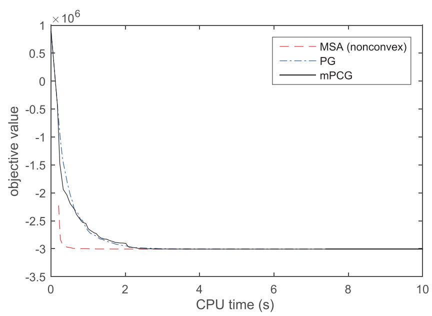
(b) $\theta = 0.1$, exemplified



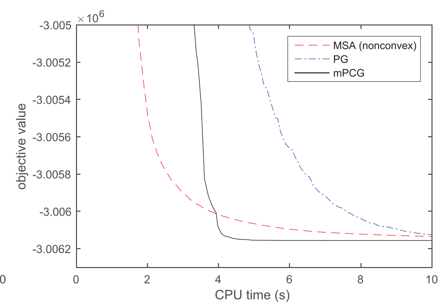
(c) $\theta = 1$, original



(d) $\theta = 1$, exemplified

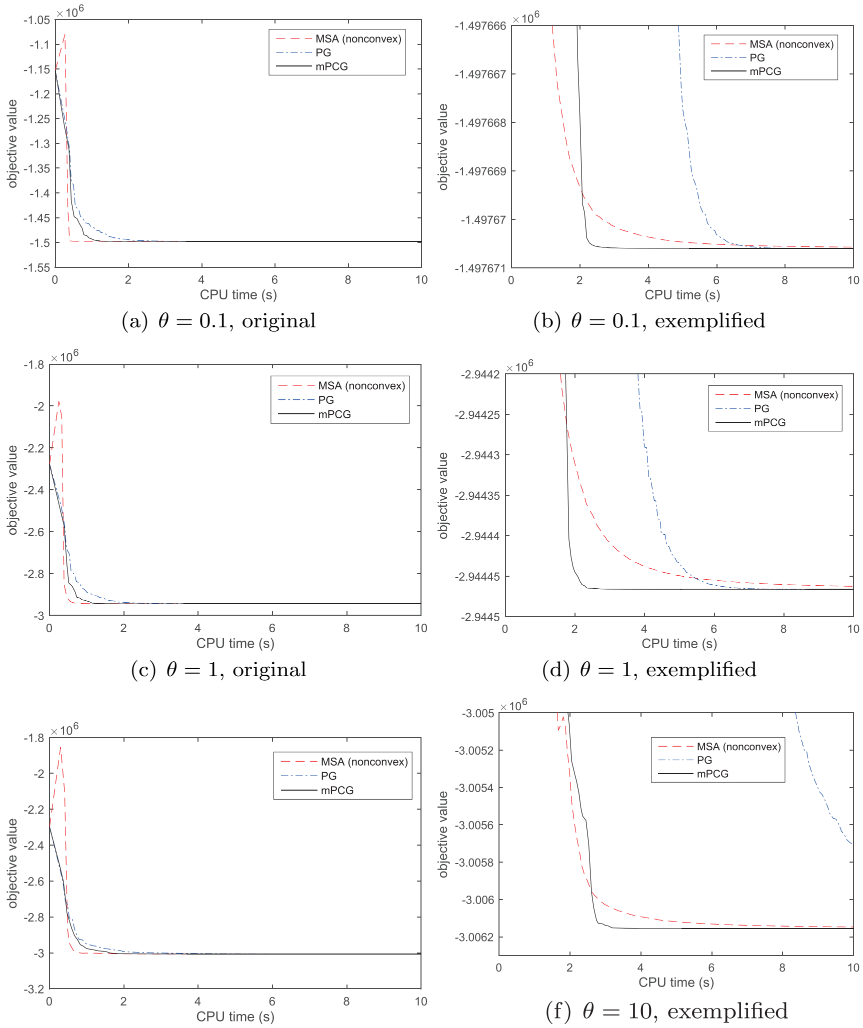


(e) $\theta = 10$, original



(f) $\theta = 10$, exemplified

Figure 2: Comparison: convex model for PG and mPCG; nonconvex model for MSA from initial point one.



(a) $\theta = 0.1$, original (b) $\theta = 0.1$, exemplified

(c) $\theta = 1$, original (d) $\theta = 1$, exemplified

(e) $\theta = 10$, original (f) $\theta = 10$, exemplified

(f) $\theta = 10$, exemplified

Figure 3: Comparison: convex model for PG and mPCG; nonconvex model for MSA from initial point two.

The above figures draws "how much decrease of the objective value is obtained in the same CPU time" for the three methods. We find the PG method is the slowest. Now we compare the mPCG method using the convex model and the MSA using the nonconvex model by setting the stopping rule as follows.

$$\frac{\|\nabla \hat{h}(\mathbf{t}^{(n)})\|}{|\mathcal{A}|} \leq \xi, \quad \text{or} \quad n \geq n_{\max} \quad \text{or} \quad \text{CPU time} \geq t_{\max}, \tag{5.1}$$

with

$$\xi = 10^{-5}, \quad n_{\max} = 1000, \quad t_{\max} = 100 \text{ seconds}.$$

Here the first term (5.1) is the relative accuracy with $|\mathcal{A}|$ being the number of links in $\mathcal{A}$.

Table 1: Comparison of MSA on convex model and mPCG on nonconvex model

	Initial point 1		Initial point 2	
$\theta = 0.1$	MSA (nonconvex)	mPCG (convex)	MSA (nonconvex)	mPCG (convex)
n_{iter}	1000	39	972	39
$\frac{\|\nabla \hat{h}(\mathbf{t}^{(n)})\|}{\|\mathcal{A}\|}$	5.42e-5	8.48e-6	1.00e-5	8.43e-6
$\hat{h}(\mathbf{t}^{(n)})$	-1497670.59	-1497670.58	-1497670.60	-1497670.58
CPU time (s)	68.69	3.01	64.41	2.90
$\theta = 1$	MSA (nonconvex)	mPCG (convex)	MSA (nonconvex)	mPCG (convex)
n_{iter}	1000	65	1000	61
$\frac{\|\nabla \hat{h}(\mathbf{t}^{(n)})\|}{\|\mathcal{A}\|}$	3.0e-4	8.09e-6	1.20e-4	8.73e-6
$\hat{h}(\mathbf{t}^{(n)})$	-2944465.87	-2944466.16	-2944466.10	-2944466.15
CPU time (s)	65.80	4.66	66.96	4.63
$\theta = 10$	MSA (nonconvex)	mPCG (convex)	MSA (nonconvex)	mPCG (convex)
n_{iter}	1000	122	1000	74
$\frac{\|\nabla \hat{h}(\mathbf{t}^{(n)})\|}{\|\mathcal{A}\|}$	3.97e-4	9.41e-6	2.06e-4	9.08e-6
$\hat{h}(\mathbf{t}^{(n)})$	-3006155.43	-3006155.91	-3006155.74	-3006155.91
CPU time (s)	67.56	9.47	69.08	5.94

Here n_{iter} in Table 1 refers to the number of iterations before it stops. We can easily see from Table 1 that the mPCG method dealing with the convex SUE programming in (3.8) needs much less iterations and CPU time to obtain a higher relative accuracy, and a lower/similar objective value, compared to the MSA for the nonconvex model in (2.16) using each initial point. All the numerical results demonstrate the usefulness of the convex SUE programming in (3.8) and the efficiency of Algorithm 4.1 for solving (3.8). We mention that the computed solution $\mathbf{x}^{(n)}$ for $\mathbf{x}^*$ is positive, and the computed solution $\mathbf{t}^{(n)}$ for $\mathbf{t}^*$ is greater than $\mathbf{t}^0$. Hence the computed $\hat{z}(\mathbf{x}^{(n)})$ and $\hat{h}(\mathbf{t}^{(n)})$ are almost the same, which coincides to our theoretical result in Remark 3.5 that $\hat{h}(\mathbf{t}^*) = \hat{z}(\mathbf{x}^*)$.

References

[1] R. P. Agdeppa, N. Yamashita and M. Fukushima, Convex expected residual models for stochastic affine variational inequality problems and its application to the traffic equilibrium problem, *Pac. J. Optim.* 6 (2010) 3-19.

[2] S. Boyd and L. Vandenberghe, Convex programming. UK: Cambridge University Press, 2004.

[3] P. H. Calamai and J. J. Moŕe, Projected gradient methods for linearly constrained problems, *Math. Program.*, 39 (1987) 93-116.

[4] X. Chen and W. Zhou, Smoothing nonlinear conjugate gradient method for image restoration using nonsmooth nonconvex minimization, *SIAM J. Imaging Sci.* 3 (2010) 765-790.

[5] S. Clark and D. Watling, Modeling network travel time reliability under stochastic demand, *Transport. Res. B-Meth.* 39 (2005) 119-140.

[6] W. H. K. Lam, H. Shao and A. Sumalee, Modeling the impact of adverse weather condition on a road network with uncertainties in demand and supply, *Transport. Res. B-Meth.* 42 (2008) 890–910.

[7] H. X. Liu, X. He and B. He, Method of successive weighted averages (MSWA) and self-regulated averaging schemes for solving stochastic user equilibrium problem, *Netw. Spat. Econ.* 9 (2009) 485–503.

[8] M. J. Maher, Stochastic user equilibrium assignment with elastic deamnd, *Traffic Enginnering & Control* 42 (2001) 163–167.

[9] M. J. Maher and P. C. Hughes, A probit-based stochastic user equilibrium assignment model, *Transport. Res. B-Meth.* 31 (1997) 341–355.

[10] Q. Meng and Z. Liu, Mathematical models and computational algorithms for probit-based asymmetric stochastic user equilibrium problem with elastic Demand, *Transportmetrica* 8 (2009) 261–290.

[11] S. Nakayama and J. Takayama, Traffic network equilibrium model for uncertain demands, In: Proceedings of the 79th Annual Meeting of the Transportation Research Board (CD-ROM), Washington, DC, 2003.

[12] Y. Sheffi, Urban Transportation Networks: Equilibrium Analysis with Mathematical Programming Methods, Englewood Cliffs: INC, 1985.

[13] Y. Sheffi and W. B. Powell, An algorithm for the equilibrium assignment problem with random link times, *Networks* 12 (2007) 191–227.

[14] A. Sumalee, R. D. Connors and P. Luathep, Network equilibrium under cumulative prospect theory and endogenous stochastic demand and supply, in *Transportation and Traffic Theory*, Lam et al. (eds.), Golden Jabilee, Springer, 2009, pp. 19–38.

[15] A. Sumalee, K. Uchida and W. H. K. Lam, Stochastic multi-modal transport network under demand uncertainties and adverse weather condition, *Transport. Res. C-Emer.* 46 (2011), 338–350.

[16] J. G. Wardrop, Some theoretical aspects of road traffic research, *ICE Proceedings: Engineering Divisions, Part II* 1 (1952) 325–362.

[17] D. Watling, A second order stochastic network equilibrium model, I: theoretical foundation, *Transport. Sci.* 36 (2002) 149–166.

[18] Q. Yu, D. Fang and W. Du, Solving the logit-based stochastic user equilibrium problem with elastic demand based on the extended traffic network model, *European J. Oper. Res.* 239 (2014) 112–118.

[19] C. Zhang, X. Chen and A. Sumalee, Robust Wardrop's user equilibrium assignment under stochastic demand and supply: Expected residual minimization approach, *Transport. Res. B-Meth.* 45 (2011) 534–552.

Manuscript received 8 August 2017
revised 18 February 2018
accepted for publication 9 June 2018

Chao Zhang
Department of Applied Mathematics, School of Science
Beijing Jiaotong University, Beijing 100044, China.
E-mail address: chzhang2@bjtu.edu.cn

Qian Zhang
Department of Applied Mathematics, School of Science
Beijing Jiaotong University, Beijing 100044, China.
E-mail address: 10121848@bjtu.edu.cn

Naihua Xiu
Department of Applied Mathematics, School of Science
Beijing Jiaotong University, Beijing 100044, China.
E-mail address: nhxiu@bjtu.edu.cn