NISS

Assessment of Parameters of a Network Microsimulator

German Molina, M.J. Bayarri, James O. Berger

Technical Report Number 133
February, 2003

National Institute of Statistical Sciences
19 T. W. Alexander Drive
PO Box 14006
Research Triangle Park, NC 27709-4006
www.niss.org

Acknowledgment and Disclaimer

This material is based upon work supported by the National Science Foundation under Grant No. DMS-00739520

Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation (NSF).

Assessment of Parameters of a Network Microsimulator

G. Molina, Duke University
M.J. Bayarri, University of Valencia
J.O. Berger, Duke University

Abstract

CORSIM is a large simulator for vehicular traffic, and is being studied with respect to its ability to successfully model and predict behavior of traffic in a 36 block section of Chicago. Inputs to the simulator include information about street configuration, driver behavior, traffic light timing, turning probabilities at each corner and distributions of traffic ingress into the system.

Data is available concerning the turning proportions in the actual neighborhood, as well as counts as to vehicular input into the system and internal system counts, during a day in May, 2000. Some of the data is accurate (video recordings), but some is quite inaccurate (observer counts of vehicles). Previous utilization of these data was to 'tune' the parameters of CORSIM – in an ad hoc fashion – until CORSIM output was reasonably close to the actual data. This common approach, of simply tuning a complex computer model to real data, can result in poor parameter choices and will completely ignore the often considerable uncertainty remaining in the parameters.

To overcome these problems, we adopt a Bayesian approach, utilizing both types of data, together with a measurement error model for the inaccurate data, to derive the posterior distribution of turning probabilities and of the parameters of the CORSIM input distribution. This posterior distribution can then be used to initialize runs of CORSIM, yielding outputs that reflect that actual uncertainty in the analysis.

Computation must be via Markov Chain Monte Carlo methodology, but this is not feasible because of the expense in running CORSIM. Hence we develop an approximation to CORSIM that can be used directly to carry out the MCMC analysis. The resulting MCMC also has some novel features based on the network structure of the problem.

Key words and phrases. CORSIM; Microsimulator; Tuning; Networks; MCMC; Fast Model Approximation.

1. Introduction

A central problem in the use of complex computer models is the calibration or inverse problem - that of determining model parameters or inputs based on field data (i.e., data from the real process being modeled). This is often done by an informal process of 'tuning,' whereby the parameters or inputs are adjusted until the model output seems to 'fit' the data.

There are several problems with such tuning. First, it is common to over-tune some parameters while under-tuning others; there is often an identifiability problem that allows the model to be tuned in many ways. Second, tuning ignores the often considerable uncertainty that exists in the tuned parameters or inputs, resulting in a potentially severe overestimation of the accuracy of model output. Third, tuning can mask the model biases that actually exist, and lead to models that are much less accurate outside the range of the observed field data.

An attractive solution to this problem is to employ Bayesian analysis to determine the posterior distribution of model parameters or inputs, given the observed field data. The resulting distribution will reflect the actual uncertainty in the parameters or inputs, and will be more resistant to over-tuning.

The chief obstacle to the Bayesian approach is computational. It is typically necessary to obtain the posterior distribution by Markov chain Monte Carlo (MCMC) methods, and this can require thousands of model runs – not feasible for many complex computer models. This difficulty can sometimes be addressed by the creation of *fast simulators* that mimic the relevant features of the model in regards to the relationship between data and inputs, and which can be analyzed directly. We give an example in this paper, involving a traffic microsimulator discussed in subsection 1.1.

1.1. Traffic Simulation

The microsimulator CORSIM (US Federal Highway administration, 1997) is a computer model of street and highway traffic. It represents individual vehicles, which enter the road network at random times, move according to local interaction rules describing governing phenomena, such as vehicle following and lane changing, and turn (or not) at intersections according to prescribed probabilities. There is inherent randomness in CORSIM: vehicles arrive at random and move randomly, albeit with rather simplified governing distributions. In the example we study, vehicles are assumed to enter the system at a given location with exponential inter-arrival times.

CORSIM is currently in wide use as the platform for a variety of traffic management and research purposes. One of these (Sacks et al., 2000) is measuring the performance of traffic signal timing plans (in cooperation with the Chicago Department of Transportation). The traffic network studied is depicted in Figure 1, and consists of a 29-intersection neighborhood in Chicago. Of specific interest will be use of CORSIM to model and analyze traffic in this network during the "rush hour" period (on a normal day).

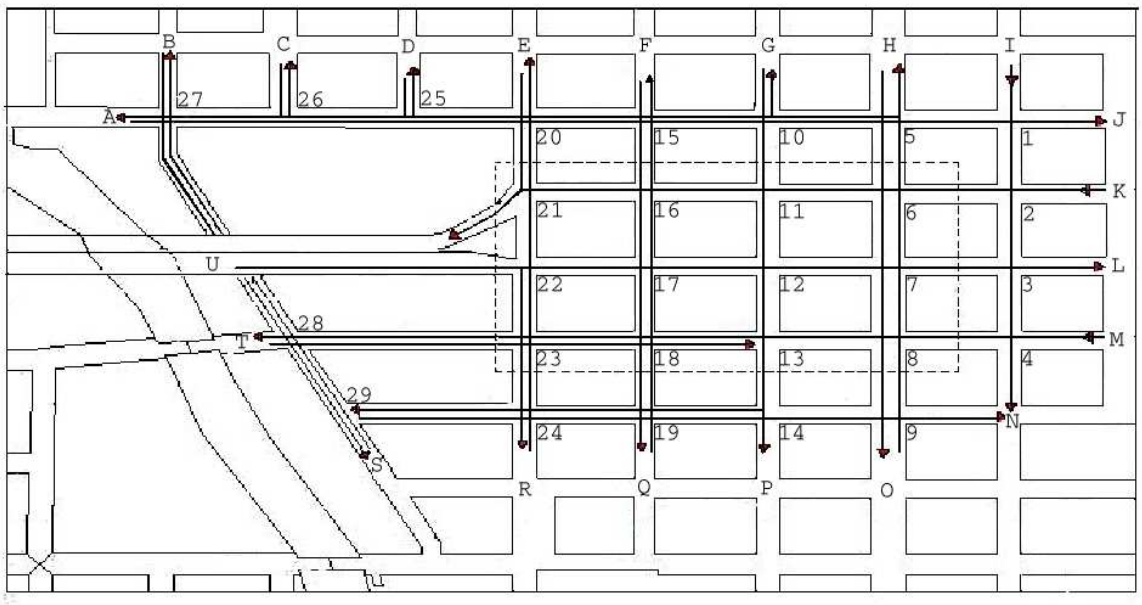

Figure 1: The Chicago traffic network being modeled by CORSIM. The dashed inner area is that for which video vehicle counts were available.

Of interest in this paper are two of the most significant (unknown) inputs to CORSIM: (i) demand and (ii) turning probabilities. Demand consists of parameters $\boldsymbol{\lambda}$ that determine the numbers of vehicles that enter the system from external streets, while the turning probabilities, $\boldsymbol{P}$, refer to the probabilities that a vehicle turns right, turns left, or goes through a given intersection. Demand and turning probabilities are street and intersection specific, so that $\boldsymbol{\lambda}$, is actually a vector of 16 numbers (for the studied system), while $\boldsymbol{P}$ is an 84-dimensional vector of probabilities. These must be determined from observational data, consisting of counts, $\boldsymbol{C}$ (subject to considerable error), made on the real-world traffic network. The available data is further discussed in Section 2.1.

1.2. Inferential Focus

The basic problem we consider is the statistical inverse problem of using the field data, $\boldsymbol{C}$, to determine the inputs, $\boldsymbol{\lambda}$, and $\boldsymbol{P}$, to the simulator; this is key before the simulator can be used to investigate possible changes to the traffic network. In the past, this has been done by tuning, as discussed in the introduction. However, there are very considerable uncertainties in the data, and these must be accounted for in the analysis.

In principle, the Bayesian approach allows accomplishment of this goal. One views the (random) simulator output as a probability distribution, given $\boldsymbol{\lambda}$, and $\boldsymbol{P}$, and relates this to the actual observations $\boldsymbol{C}$ through a measurement error model. Specifying a prior distribution for $\boldsymbol{\lambda}$, and $\boldsymbol{P}$ then allows use of Bayes theorem to obtain their posterior distribution, given the data $\boldsymbol{C}$, to be denoted by $\pi(\boldsymbol{\lambda}, \boldsymbol{P} \mid \boldsymbol{C})$. This distribution automatically incorporates all the uncertainty in the simulator inputs $\boldsymbol{\lambda}$, and $\boldsymbol{P}$. The challenge of determining $\pi(\boldsymbol{\lambda}, \boldsymbol{P} \mid \boldsymbol{C})$ is discussed in the next section.

Uncertainty in simulator predictions can then be assessed by treating $\pi(\boldsymbol{\lambda}, \boldsymbol{P} \mid \boldsymbol{C})$ as the "random input distribution" for the simulator, and making repeated runs of the simulator, initialized by draws from this distribution. This is not significantly more expensive than running the basic simulator in our situation; it is, in any case a stochastic simulator so its predictions can only be ascertained through repeated runs, and starting each run with $\boldsymbol{\lambda}$ and $\boldsymbol{P}$ chosen from $\pi(\boldsymbol{\lambda}, \boldsymbol{P} \mid \boldsymbol{C})$ is virtually as cheap as starting each run with $\boldsymbol{\lambda}$ and $\boldsymbol{P}$ fixed.

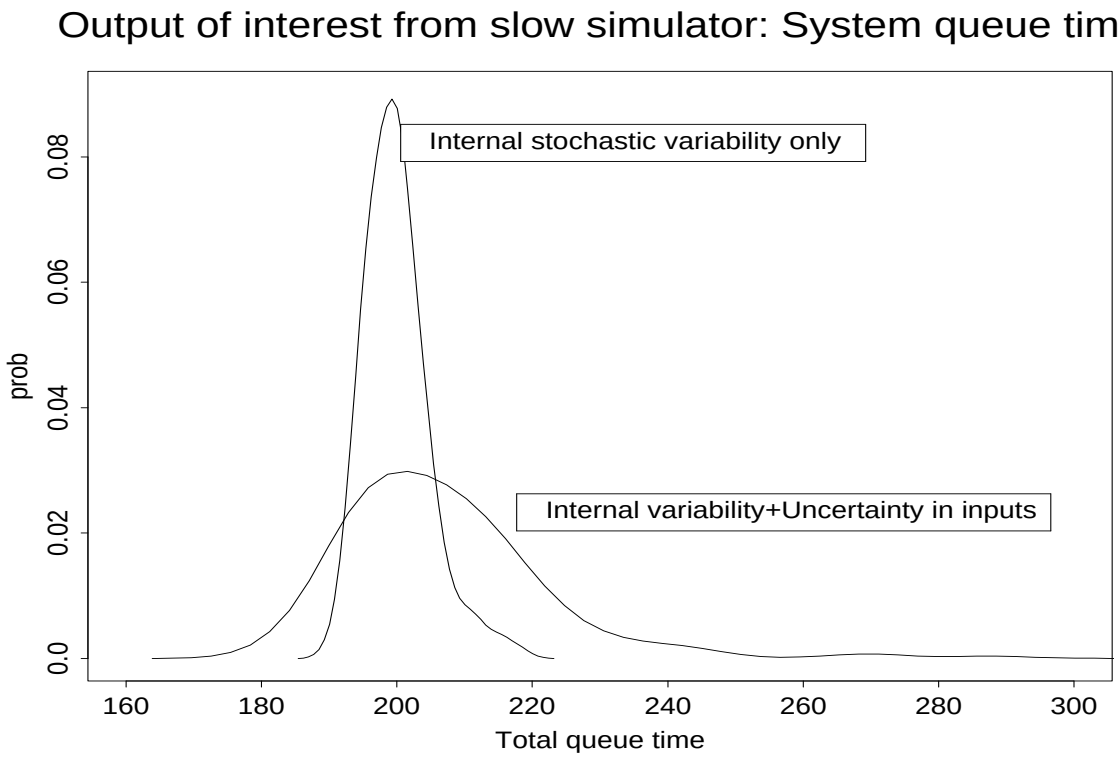

Figure 2: Total queueing time of 'tuned' CORSIM and Bayes CORSIM.

As an illustration of the importance of incorporating this uncertainty in the initialization of CORSIM, Figure 2 presents the distribution of a key output feature of CORSIM, the total queueing time of vehicles in the network. 'Tuned CORSIM' refers to this output distribution when the tuned version of CORSIM was used. 'Bayes CORSIM' refers to use of the Bayesian posterior input distribution to CORSIM. Clearly the distribution when the uncertainty in inputs is acknowledged is markedly more diffuse, and could lead to very different policy decisions. This thus provides dramatic evidence of the dangers of simply tuning a model.

1.3. The Fast Simulator

A single run of CORSIM, on a typical PC platform and in the setting discussed in Section 1.1, typically takes 2 to 3 minutes. While much faster than many complex computer models, this is still too slow to use the simulator directly to obtain $\pi(\boldsymbol{\lambda}, \boldsymbol{P} \mid \boldsymbol{C})$. The reason is that the only available method for direct determination of the posterior is the Markov chain Monte Carlo (MCMC) approach (see, for instance, Chen et al., 2000, Robert and Casella, 1999). In our problem, however, there are upwards of 200 unknown and highly dependent parameters under analysis, and 2-3 minute simulator run-times will not allow an MCMC analysis in a situation of such complexity.

We proceed, therefore, by creating a simpler stochastic network that mimics the traffic simulator, with respect to the key features $\boldsymbol{\lambda}$ and $\boldsymbol{P}$ under study. This simpler network omits many other features of CORSIM, such as vehicle waiting times, but is arguably accurate in terms of its representation of the effect of $\boldsymbol{\lambda}$ and $\boldsymbol{P}$. One can then determine the posterior distribution, $\pi^*(\boldsymbol{\lambda}, \boldsymbol{P} \mid \boldsymbol{C})$, of $\boldsymbol{\lambda}$ and $\boldsymbol{P}$ in the simpler network, and use this as the posterior for CORSIM.

2. Data and Measurement Error

2.1. The Observed Vehicle Counts

We identify each intersection in the Chicago Network (Figure 1) with numbers. Intersections just outside the network are also identified with letters. The right part of the network will be repeatedly used for demonstration. It is shown in Figure 3 and it consists of intersections 1 through 8. All movements in and out of intersections 6,7,8 are recorded in video. Intersections I through N will also be needed to identify inputs to (demands) and outputs from the system. In what follows, this subnet will be denoted simply by RN (Restricted Network)

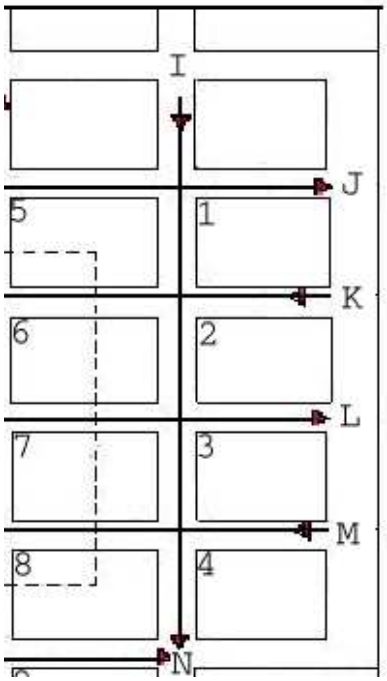

Figure 3: The restricted network (RN) used for illustration. It consists of the rightmost part of the network in Figure 1.

In the following notation, the subindices refer to a number or a letter identifying an intersection. A dot will indicate summation over the corresponding index.

The data, $\boldsymbol{C}$, is a vector of counts of vehicles. Let C_{ijk} denote the (*observed*) count of vehicles at intersection j (any of the 29 intersections), arriving from intersection i and going to intersection k. Thus, for instance C_{348} is the count of vehicles arriving at intersection 4 from intersection 3, and turning right towards intersection 8. It is also convenient to define $C_{ij\bullet} = \sum_k C_{ijk}$; this represents the total observed number of vehicles entering intersection j from intersection i. The counts fall into three classes of data:

DEMAND COUNTS:

These are counts, made over a one-hour period, by observers placed on the streets entering the traffic neighborhood in Figure 1, and correspond to certain of the $C_{ij\bullet}$ above. In particular, if i denotes an outside node (identified with a letter in our net), then $C_{ij\bullet}$ are the observed demand counts entering the system from external node i (notice that in this network, there is only one input to the system from any of the external nodes). Thus, for instance, $C_{M4\bullet}$ are the cars entering the system from M and into node 4. Some of these counts are suspected of being

quite inaccurate, with errors potentially as high as 50%. We represent all these counts by $\boldsymbol{C}^D$, and let

$$I^D = \{\text{set of indices corresponding to external input (demand) nodes}\}; \tag{2.1}$$

Hence, $\boldsymbol{C}^D = \{C_{ij\bullet},\ \text{for } i \in I^D\}$. For example, if one only considered RN, I^D would consist of I, K, M and 5 (which will be considered an external node for the purposes of illustrations in the RN), and $\boldsymbol{C}^D = \{C_{I1\bullet}, C_{K2\bullet}, C_{M4\bullet}, C_{51\bullet}\}$.

VIDEO COUNTS:

At the intersections in Figure 1 that lie within the central dashed rectangle, video cameras were placed that recorded all vehicles passing through the intersection over the one-hour period. The recordings were later analyzed to exactly determine the numbers of vehicles that go left, right and through at each of these intersections. We denote all these intersections by I^V, so that

$$I^V = \{\text{set of indices in the video area}\}, \tag{2.2}$$

and we denote the corresponding counts by $\boldsymbol{C}^V$. These counts can be treated as exact.

TURNING COUNTS:

These are counts, made by observers over shorter time intervals (typically between 10 and 20 minutes), of the numbers of vehicles that go left, right and through at each intersection not in the inside (video) area. We represent all these counts by $\boldsymbol{C}^T$, and let

$$I^T = \{\text{set of indices of intersections with manual turning counts}\}; \tag{2.3}$$

Also, for every intersection j (either in I^T or I^V), define the *ancestors* of node j, A_j as

$$A_j = \{\text{indices } i \text{ such that cars can go from } i \text{ to } j\ \}, \tag{2.4}$$

and for each i in A_j define the descendants of the *link* $i\,j$, D_{ij}, as

$$D_{ij} = \{\text{indices } k \text{ such that the } \textit{route} \text{ ijk is possible}\}. \tag{2.5}$$

Then $\boldsymbol{C}^T = \{C_{ijk},\ \text{for } j \in I^T, i \in A_j, k \in D_{ij}\}$. For example, in RN, $I^T = \{1, 2, 3, 4\}$, and, for instance, $A_2 = \{1, K\}$, $D_{K2} = \{6, 3\}$; $\boldsymbol{C}^T = (C_{I1J},\ C_{I12},\ C_{51J},\ C_{512},\ C_{126},\ C_{123},\ C_{K26},\ C_{K23},\ C_{23L},\ C_{234},\ C_{73L},\ C_{734},\ C_{348},\ C_{34N},\ C_{M48},\ C_{M4N})$.

2.2. Latent Counts and Measurement Error

Key basic elements of the stochastic network are numbers N_{ijk}, that correspond to the true numbers of vehicles passing through the traffic network for the entire observational time period; these are defined using the same notational convention as that for the C_{ijk}. Note, however, that the turning counts were defined over shorter time intervals, and the corresponding true numbers of vehicles for these shorter periods will be denoted by $\widetilde{N}_{ijk}$. Also define $\widetilde{\boldsymbol{N}}^T$ as the collection of $\widetilde{N}_{ijk}$ for $j \in I^T$.

MEASUREMENT ERROR FOR DEMAND COUNTS:

It is assumed that the $C_{ij\bullet}$, arising from the streets entering the traffic neighborhood ($i \in I^D$), follow a Poisson distribution with mean $b_{ij}\, N_{ij\bullet}$, where $b_{ij} > 0$ and $b_{ij} - 1$ is the unknown mean proportional bias of the observer counting the corresponding $C_{ij\bullet}$. Thus

$$C_{ij\bullet} | \, b_{ij}, N_{ij\bullet} \sim \text{Po}(C_{ij\bullet} \mid b_{ij}\, N_{ij\bullet}), \quad \text{for } i \in I^D.$$

Recalling that $\boldsymbol{C}^D$ is the vector of these incoming $C_{ij\bullet}$, and $\boldsymbol{N}^D$ and $\boldsymbol{b}$ are the corresponding vectors of $N_{ij\bullet}$ and b_{ij}, and assuming that the counts $C_{ij\bullet}$ are independent, we have thus specified the density

$$f_1(\boldsymbol{C}^D \mid \boldsymbol{N}^D, \boldsymbol{b}) = \prod_{i \in I^D} \text{Po}(C_{ij\bullet} \mid b_{ij}\, N_{ij\bullet}). \tag{2.6}$$

A key feature of this measurement error model is that the variance is proportional to the true latent vehicle count, a reasonable assumption. Also, there is only one parameter b_i at each location, helpful since information about the biases is quite limited (and indirect). A variety of other error models were considered (e.g., discrete normal with a variety of heteroscedastic variance structures), but the simple Poisson model seemed most effective in explaining the data.

ASSUMPTION FOR VIDEO COUNTS AND TURNING COUNTS:

We will assume that the counts obtained from the video cameras and the manually collected turning counts are exact, i.e., have no measurement error. Thus we assume that $\boldsymbol{C}^V = \boldsymbol{N}^V$ and $\boldsymbol{C}^T = \widetilde{\boldsymbol{N}}^T$. This is essentially correct for the video counts, but is considerably more questionable for the turning counts. Indeed, for the turning counts, this assumption is primarily made for pragmatic reasons, including the following:

- It is not unreasonable to assume that the biases at a given intersection for the various turning movements are the same, so that the observed turning proportions would be correct. Any systematic bias would then only affect the posterior variances, a considerably

less influential error.

- Because of the short time period (at most twenty minutes) over which the turning counts were obtained, the data will naturally lead to comparatively vague posterior distributions on the turning probabilities, so it is not so compelling to introduce additional measurement error.

- There is not enough data to effectively estimate numerous turning probability biases, and the stability of the resulting MCMC would be in question if this were attempted.

3. Analysis Using CORSIM

We first discuss the analysis that had previously been performed with the data using CORSIM, based on ad hoc tuning. Then we discuss a potential full Bayesian analysis, to clarify the difficulty that is involved and set the stage for the need for a network approximation (implemented in the next section).

3.1. Adhoc Analysis

The traditional way of 'fitting' complex computer models involves some type of tuning of the parameters of the models, based on the data. For CORSIM and the data described above, this was done in two stages. The first stage consisted of a trial and error process of adjusting the manual counts until they seemed to be roughly compatible with the counts from the video data. These 'adjusted counts' were then used to fit the parameters $\boldsymbol{\lambda}$ and $\boldsymbol{P}$, partly by moment methods and partly by trial-and-error using CORSIM runs. As an example, the triangles in Figure 4 indicate the resulting estimates for two of the λ_{ij}.

As mentioned in the introduction, ad hoc tuning has several deficiencies. Primary among them are that it can lead to inferior estimates, and that it does not readily allow assessment of the uncertainty that exists in the parameter estimates, typically leading to considerable overconfidence in use of the model. One potential method for overcoming these deficiencies is to implement a Bayesian approach.

3.2. A Full Bayesian Analysis

A Bayesian analysis requires the likelihood, a prior distribution for unknown parameters, and computational ability to deal with the posterior distribution. The first two are, in principle, available here.

THE CORSIM LIKELIHOOD:

CORSIM is a stochastic microsimulator so that, even with fixed initial parameter values $(\boldsymbol{\lambda}, \boldsymbol{P})$, it will produce random outputs. Of interest here are the latent vehicle counts at each link. We will denote by $f_C(\boldsymbol{N}^V, \boldsymbol{N}^D, \boldsymbol{N}^T, \widetilde{\boldsymbol{N}}^T \mid \boldsymbol{\lambda}, \boldsymbol{P})$ the density, given $\boldsymbol{\lambda}$ and $\boldsymbol{P}$, of these latent counts arising as random outputs from CORSIM. Note that this is not available in closed form, and can only be estimated by repeatedly running the microsimulator.

Recall that the data consists of the video, demand and turning counts, $\boldsymbol{C}^V$, $\boldsymbol{C}^D$, and $\boldsymbol{C}^T$, respectively. The measurement error density for the demand counts, given $\boldsymbol{N}^D$, was defined in (2.6). We also made the assumptions that $\boldsymbol{C}^V = \boldsymbol{N}^V$ and $\boldsymbol{C}^T = \widetilde{\boldsymbol{N}}^T$, so that we can write the CORSIM likelihood of the data as

$$f_C(\boldsymbol{C}^V, \boldsymbol{N}^D, \boldsymbol{N}^T, \boldsymbol{C}^T \mid \boldsymbol{\lambda}, \boldsymbol{P})\, f_1(\boldsymbol{C}^D \mid \boldsymbol{N}^D, \boldsymbol{b}). \tag{3.7}$$

THE PRIOR DISTRIBUTION:

We must specify the prior distributions for $\boldsymbol{\lambda}$, $\boldsymbol{P}$, and $\boldsymbol{b}$. We (independently) utilize the standard objective prior distributions $\lambda_{ij}^{-1/2}$, $i \in I_D$ for the Poisson means and assign the turning probabilities, for a given $\boldsymbol{P_{ij}}$, the Jeffreys prior proportional to $\prod_{k \in D_{ij}} P_{ijk}^{-1/2}$ (where, of course, $\sum_{k \in D_{ij}} P_{ijk} = 1$); the resulting joint density is then

$$\pi_C(\boldsymbol{\lambda}, \boldsymbol{P}) = \left(\prod_{i \in I_D} \lambda_{ij}^{-1/2}\right) \left(\prod_{j \in I_T \cup I_V,\ i \in A_j,\ k \in D_{ij}} P_{ijk}^{-1/2}\right). \tag{3.8}$$

We model the b_{ij} as being i.i.d. from a Gamma(α, β) distribution, with α and β having a constant prior distribution on the region $0 < \alpha < 2\beta$ (so that the mean bias is restricted to be less than 100%, a rather mild restriction). This joint prior density is thus

$$\pi_1(\boldsymbol{b}, \alpha, \beta) = \prod_{i \in I_D} \mathrm{Ga}(b_{ij} | \alpha, \beta) 1_{0 < \alpha < 2\beta}. \tag{3.9}$$

POSTERIOR COMPUTATION:

The posterior distribution is simply proportional to the product of the likelihood and the priors defined above. Note that this is not available analytically, nor are any conditional posterior distributions available. Hence the only realistic method of proceeding would be to do an MCMC analysis based on some variant of a Metropolis-Hastings scheme. Any such scheme requires, at each iteration, computation of f_C. Since this density is itself not available analytically, it would have to be estimated (at given values of the parameters) by making repeated (typically hundreds or thousands) of runs of CORSIM. At 2 to 3 minutes per run, this means that several hours (or

days) would be required to make one Metropolis iteration. Since we have a very high dimensional parameter space (with highly dependent parameters), it is apparent that this type of analysis is not going to be implementable, especially on a routine basis, for traffic models. This motivates the development of the fast network approximation in the next section.

4. Development and Analysis of the Simulator Approximation

Since $f_C(\boldsymbol{C}^V, \boldsymbol{N}^D, \boldsymbol{N}^T, \boldsymbol{C}^T \mid \boldsymbol{\lambda}, \boldsymbol{P})$ is not readily computable, we develop a computable approximation which can be viewed as creating a stochastic simulator or stochastic network (see Cowell, et.al., 1999) that approximates the key features of the density of the latent counts. The key idea behind the approximation is to assume that cars pass through the network instantaneously, but have the same input rates and turning probabilities as CORSIM itself. This, of course, misses many details of CORSIM having to do with waiting times, etc., but is a plausible approximation to the actual distribution of vehicle counts. Note that we are trying to approximate CORSIM in its steady state, so that elimination of the transit times, should not strongly affect the distribution of vehicle counts.

For this instantaneous version of CORSIM, the corresponding approximation to the joint distribution $f_C(\boldsymbol{C}^V, \boldsymbol{N}^D, \boldsymbol{N}^T, \boldsymbol{C}^T \mid \boldsymbol{\lambda}, \boldsymbol{P})$ can be factored as follows:

$$f_2(\boldsymbol{N}^D \mid \boldsymbol{\lambda})\ f_3(\boldsymbol{C}^T \mid \boldsymbol{N}^D, \boldsymbol{N}^T)\ f_4(\boldsymbol{C}^V, \boldsymbol{N}^T \mid \boldsymbol{N}^D, \boldsymbol{P}). \tag{4.10}$$

This is justified below, and the form of each of these densities is derived.

Density $f_2(\boldsymbol{N}^D \mid \boldsymbol{\lambda})$ for demand:

Vehicles enter the CORSIM network randomly from 'outside,' with exponential inter-arrival rates λ_{ij}. Since $N_{ij\bullet}$, for $i \in I^D$, is the number of vehicles arising from the exponential inter-arrival rate λ_{ij} over a one-hour period, it is Poisson with mean λ_{ij}:

$$N_{ij\bullet} | \lambda_{ij} \sim \text{Po}(N_{ij\bullet} \mid \lambda_{ij}) \quad \text{for} \quad i \in I^D.$$

In the 'instantaneous' version of CORSIM that we are employing, the entering vehicles are independent of anything else in the system (and of each other), and hence we have that

$$f_2(\boldsymbol{N}^D \mid \boldsymbol{\lambda}) = \prod_{i \in I^D} \text{Po}(N_{ij\bullet} \mid \lambda_{ij}), \tag{4.11}$$

where recall that $\boldsymbol{N}^D$ refers to the vector of incoming $N_{ij\bullet}$, for $i \in I^D$, and $\boldsymbol{\lambda}$ is the vector of

the λ_{ij} for $i \in I^D$.

DENSITY OF $f_3(\boldsymbol{C}^T \mid \boldsymbol{N}^D, \boldsymbol{N}^T)$ FOR TURNING COUNTS:

Recall that $\widetilde{N}_{ijk}$ is the number of vehicles that travel from i to j go to k over the 'short' period of time. Denote by $\widetilde{\boldsymbol{N}}_{ij}$ the vector with components $\widetilde{N}_{ijk}$ for $k \in D_{ij}$. The distribution of $\widetilde{\boldsymbol{N}}_{ij}$ given all the relevant N_{ijk}'s (and the rest of parameters) depends only on the N's, specifically, on the total number of vehicles arriving to j from i in the short period of time ($\widetilde{N}_{ij\bullet} = \sum_{k \in D_{ij}} \widetilde{N}_{ijk}$), the total number arriving in the 1-hour total period ($N_{ij\bullet} = \sum_{k \in D_{ij}} N_{ijk}$), and the number among these going to the different directions (to be similarly denoted by $\boldsymbol{N}_{ij}$). Under stationarity, the distribution of $\widetilde{\boldsymbol{N}}_{ij}$ is a multivariate hypergeometric distribution with parameters $N_{ij\bullet}, \boldsymbol{N}_{ij}$ and $\widetilde{N}_{ij\bullet}$. This can be approximated by a multinomial distribution with parameters $\widetilde{N}_{ij\bullet}$ and probabilities $\boldsymbol{N}_{ij}/N_{ij\bullet}$. Finally, since the numbers N_{ijk} are large in our study, we can approximate $N_{ijk}/N_{ij\bullet} \approx P_{ijk}$, the true proportion of vehicles going to k from the link ij. Therefore:

$$\widetilde{\boldsymbol{N}}_{ij} \mid \boldsymbol{N}^D, \boldsymbol{N}^T \approx \widetilde{\boldsymbol{N}}_{ij} \mid \widetilde{N}_{ij\bullet}, \boldsymbol{P}_{ij} \sim \text{Mul}(\widetilde{\boldsymbol{N}}_{ij} \mid \widetilde{N}_{ij\bullet}, \boldsymbol{P}_{ij}).$$

Equating $\widetilde{N}_{ijk} = C_{ijk}$ for $j \in I^T$ (see the end of Subsection 2.2.), and noticing that $\widetilde{N}_{ij\bullet} = C_{ij\bullet}$ is the observed sum of the vehicles going to the different directions from j, and hence is known, we can conclude that

$$f_3(\boldsymbol{C}^T \mid \boldsymbol{N}^D, \boldsymbol{N}^T) \approx \prod_{j \in I_T, i \in A_j} \text{Mul}(\widetilde{\boldsymbol{C}}_{ij} \mid \widetilde{C}_{ij\bullet}, \boldsymbol{P}_{ij}). \tag{4.12}$$

DENSITY OF $f_4(\boldsymbol{N}^V, \boldsymbol{N}^T \mid \boldsymbol{N}^D, \boldsymbol{P})$ FOR THE LINK VEHICLE TOTALS:

CORSIM specifies that vehicles turn at intersections independently, which we thus also assume in the instantaneous model. A single vehicle approaching intersection $j \in I^T \cup I^V$ (i.e., those intersections other than the input nodes) will thus turn (or go straight) according to the $\text{Mul}(\boldsymbol{P_{ij}}, 1)$ distribution. Since the vehicles are independent, these multinomial densities for each vehicle at each intersection can simply be multiplied to obtain the joint density of a single realization of all vehicles passing through the system, given the initialization, $\boldsymbol{N}^D$, of vehicles entering the system.

To determine the density of $\boldsymbol{N}^T$, given $\boldsymbol{N}^D$ and $\boldsymbol{P}$, one simply counts up the number of ways vehicles can pass through the system to yield a given $\boldsymbol{N}^T$. At intersection j, with $N_{ij\bullet} = \sum_{k \in D_{ij}} N_{ijk}$ being the number of vehicles entering from i, the number of ways that the vehicles can result in turning (or straight) numbers N_{ijk} is simply the corresponding multinomial

coefficient. Taking the product of these multinomial coefficients over all intersections, completes the density specification. Interestingly, this density is thus proportional to the density of the product of individual multinomial distributions at each intersection, i.e., letting $\boldsymbol{N}_{ij}$ denote the vector with components N_{ijk} with $k \in D_{ij}$, it is the product of the densities

$$\boldsymbol{N_{ij}} \mid \boldsymbol{P_{ij}} \ \sim \ \mathrm{Mul}\,(\,\boldsymbol{N_{ij}} \mid \boldsymbol{P_{ij}},\ N_{ij\bullet}) \quad \text{for} \ \ j \in I_T \cup I_V,\, i \in A_j\,.$$

It is important to realize that there are numerous constraints among the N_{ijk} and $N_{ij\bullet}$ in our network approximation. For instance, the total number of vehicles entering an intersection is equal to the number leaving the intersection. Thus the actual density of the vehicle numbers is the product of the above multinomials, but with the constraints inserted. Furthermore (and of crucial effect), the video counts lead to known values of some of the $N_{\bullet jk}$ (for $j \in I_V$) and some of the $N_{ij\bullet}$ (for $i \in I_V$), and these known values induce other constraints. Letting $\mathcal{N}$ denote the region implied by all these constraints, and $1_{\mathcal{N}}$ denote the indicator function on this region, the final density of the link and turning numbers is

$$f_4(\boldsymbol{N}^V, \boldsymbol{N}^T \mid \boldsymbol{N}^D, \boldsymbol{P}) \propto \prod_{j \in I_T \cup I_V,\, i \in A_j} \mathrm{Mul}\,(\,\boldsymbol{N_{ij}} \mid \boldsymbol{P_{ij}},\ N_{ij\bullet})\ 1_{\mathcal{N}}. \tag{4.13}$$

The handling of these constraints and the systematic identification of the region $\mathcal{N}$ is deferred to Subsection 5.1.

THE POSTERIOR DISTRIBUTION:

By Bayes theorem, the posterior distribution $\pi^*(\boldsymbol{N^V, N^T, N^D}, \boldsymbol{\lambda, P, b, \alpha, \beta} \mid \boldsymbol{C})$, of all unknowns given the data $\boldsymbol{C}$, is simply proportional to the product of the likelihood and the prior, i.e.

$$f_1(\boldsymbol{C}^D \mid \boldsymbol{N}^D, \boldsymbol{b}) f_2(\boldsymbol{N}^D \mid \boldsymbol{\lambda}) f_3(\boldsymbol{C}^T \mid \boldsymbol{N}^D, \boldsymbol{N}^T) f_4(\boldsymbol{N}^V, \boldsymbol{N}^T \mid \boldsymbol{N}^D, \boldsymbol{P}) \pi_C(\boldsymbol{\lambda}, \boldsymbol{P}) \pi_1(\boldsymbol{b}, \alpha, \beta),$$

where the distributions are given in (2.6), (4.11), (4.12), (4.13), (3.8), and (3.9), respectively. Although $\pi(\boldsymbol{\lambda}\,, \boldsymbol{P}, \boldsymbol{b}, \alpha, \beta)$ was improper, it can be shown that this posterior distribution is proper.

5. MCMC Computation

The chief difficulty in the analysis is dealing with the constraints specified by $1_{\mathcal{N}}$. Indeed, the most important step in the computation is to effect a reparameterization of the N_{ijk} so that

unneeded variables are eliminated and the constraints take a simple form. Indeed, we were unsure that an effective MCMC analysis could be implemented here, until a scheme for carrying out this reparameterization was found. (In principle, one could simply numerically compute the constraint at each step of the MCMC procedure via linear programming, but this would be too time consuming because of the large number of iterations that are needed.)

5.1. Reparameterization: Handling the Restrictions

There are four types of restrictions to be imposed on the posterior distribution:

R1. *Turning probability restrictions.* The turning probabilities at any intersection must add up to one. This can easily be introduced in the likelihood by expressing one of the P's in terms of the other(s) at each intersection.

R2. *Nonnegativity restrictions.* Every count must be a nonnegative integer. This trivial restriction becomes essential when sampling the N's in the MCMC.

R3. *Physical restrictions.* In the instantaneous model, the total number of cars entering a link coming from any intersection must equal the number of cars in that link. Therefore for any j, k, we have that $N_{\bullet jk} = \sum_i N_{ijk}$. Similarly $N_{ij\bullet} = \sum_k N_{ijk}$. Note that $N_{ij\bullet} = N_{\bullet ij}$.

R4. *Video restrictions.* All of the $N_{\bullet jk}$ and $N_{ij\bullet}$ in the video zone of the fast simulator are observed and hence impose restrictions on the non-observed N's. Thus, for example, it was recorded that the number of cars going from intersection 4 to intersection 8 is 740. Hence, $N_{\bullet 48} = 740 = N_{348} + N_{M48}$.

Although restrictions R1 and R2 are easy to handle, restrictions R3 and R4 are not. Combined, they form a set of linear equations where the number of unknowns is larger than the number of equations. For instance, in the small network (see Figure 3), the restrictions (R3 and R4) are as follows:

physical restrictions

$$\begin{aligned} N_{12\bullet} &= N_{I12} + N_{51\bullet} - N_{51J} \\ N_{23\bullet} &= N_{123} + N_{K2\bullet} - N_{K26} \\ N_{34\bullet} &= N_{234} + N_{73\bullet} - N_{73L} \end{aligned}$$

video restrictions

$$
\begin{aligned}
N_{\bullet 26} &= N_{K26} + N_{12\bullet} - N_{123} \\
N_{\bullet 48} &= N_{M48} + N_{34\bullet} - N_{34N}
\end{aligned} \tag{5.14}
$$

In general, the m restrictions R3 and R4 define a set of m linear equations in the N's, with k ($m < k$) unknown variables, and m known (one for each equation). We write this set in matrix form as follows:

$$
\boldsymbol{\Gamma}\ \boldsymbol{N} = \boldsymbol{\Lambda}, \tag{5.15}
$$

where $\boldsymbol{\Gamma}$ is a $m \times k$ matrix of {-1,0,1} coefficients associated with the restrictions, $\boldsymbol{N}$ is a $k \times 1$ vector of unknowns, and $\boldsymbol{\Lambda}$ is an $m \times 1$ vector of known (observed) values. For instance, in the small network (RN), the physical and video restrictions (5.14) have $m = 5, k = 13$, and can be written in the form (5.15) whith

$$
\boldsymbol{\Gamma} = \begin{pmatrix}
1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 \\
0 & 0 & 0 & 1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 1 & 0 & 0 & 0 & 0 \\
-1 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -1 & 1 & 0 & 0
\end{pmatrix},
$$

$\boldsymbol{N} = (N_{12\bullet}, N_{I12}, N_{51J}, N_{K26}, N_{123}, N_{23\bullet}, N_{34\bullet}, N_{234}, N_{73L}, N_{M48}, N_{34N}, N_{51\bullet}, N_{K2\bullet})'$,
$\boldsymbol{\Lambda} = (0, 0, N_{73\bullet}, -N_{\bullet 26}, -N_{\bullet 48})'$.

The proposed reparameterization is then as follows: find any $m \times m$ non-singular partition $\boldsymbol{\Gamma}^*$ of $\boldsymbol{\Gamma}$ (guaranteed to exist because of the nontriviality of the restrictions). For any such partition there exists a matrix $\boldsymbol{R}$ (the remaining columns of $\boldsymbol{\Gamma}$ not in $\boldsymbol{\Gamma}^*$) such that we can rewrite (5.15) as $\boldsymbol{\Gamma}^*\boldsymbol{N}^* + \boldsymbol{R}\boldsymbol{N}_{\boldsymbol{R}} = \boldsymbol{\Lambda}$. Then

$$
\boldsymbol{N}^* = (\boldsymbol{\Gamma}^*)^{-1}\boldsymbol{\Lambda} - (\boldsymbol{\Gamma}^*)^{-1}\boldsymbol{R}\boldsymbol{N}_{\boldsymbol{R}} \tag{5.16}
$$

provides the required reparameterization. Note that neither $\boldsymbol{\Gamma}^*$ nor $\boldsymbol{N}^*$ are typically unique.

An easy way to find $\boldsymbol{\Gamma}^*$ is to compute the reduced echelon form of the matrix $\boldsymbol{\Gamma}$ and select the m columns that span its column space. In RN, it is formed by the columns 1,2,4,7 and 8.

After the reparameterization, one can proceed by Gibbs sampling (see Chen et al., 2000, Robert and Casella, 1999). The full conditional distributions of the λ_{ij}, $\boldsymbol{P_{ij}}$, $\boldsymbol{b}$, and β are Gamma, Dirichlet, Gamma and restricted Gamma, respectively, with easily specified parameters (see the Appendix). The full conditional density of α is log-concave and so can be sampled by straightforward rejection sampling.

The full conditionals of the various N have no simple form, but are discrete distributions over specified ranges. While, in principle, they can thus be directly sampled, in practice this can only be done effectively after two significant innovations. First a method for determining the constraints must be found which is considerably faster than straight linear programming. Second, it is far from trivial to find legitimate starting values for the latent vehicle numbers in the MCMC. Indeed, this is related to solution of a system of restricted diophantine linear equations, which has an extensive literature in mathematics and computer sciences. The details of these two needed innovations are given in the appendix.

5.2. Utilizing the MCMC output

The output of the MCMC analysis is a large sample of (dependent) realizations from the posterior distribution. For use in CORSIM, one thus simply records this sample of ($\boldsymbol{\lambda}$,$\boldsymbol{P}$), and uses the sample directly as the input values for the exercise of CORSIM. In practice, only a few hundred values of the ($\boldsymbol{\lambda}$,$\boldsymbol{P}$) vectors will typically be utilized in a CORSIM prediction, whereas the MCMC runs will typically result in, say, 100,000 realizations of ($\boldsymbol{\lambda}$,$\boldsymbol{P}$) (because of the considerable dependence between iterations of the MCMC). Hence, one need typically only save, say, every 500th realization of ($\boldsymbol{\lambda}$,$\boldsymbol{P}$) from the MCMC run; the resulting 200 realizations will then also be much less dependent, desirable when used as the inputs for CORSIM. For an indication of the extent of mixing in the Markov chain, refer to Figure 4, where two typical plots of the MCMC runs are given. The mixing is clearly somewhat slow, but seems adequate for the situation.

Figure 2 already indicated the type of primary use of this MCMC output. It is interesting, however, to look at certain other aspects of the posterior, to illustrate various points that have been made in the article.

Figure 4 presents histograms of posterior values of the λ_{ij} for two adjacent incoming streets, along with their fitted values obtained in the earlier ad hoc analysis for the actual observed counts (indicated by the vertical lines) and 'tuned' counts (indicated by the triangles). Note first that the posterior distributions of the λ_{ij} are considerably shifted from the values that would have been obtained by fitting to the raw counts. The tuning attempted to correct the clear bias that was in the data concerning the number of vehicles entering the network, but note what happened: $\lambda_{F,15}$ was over-tuned, while $\lambda_{E,20}$ was under-tuned. As these were adjoining streets, this corrected the most egregious bias problem, but ended up replacing one big problem by two smaller problems. This is a common occurrence with ad hoc tuning; it is far easier

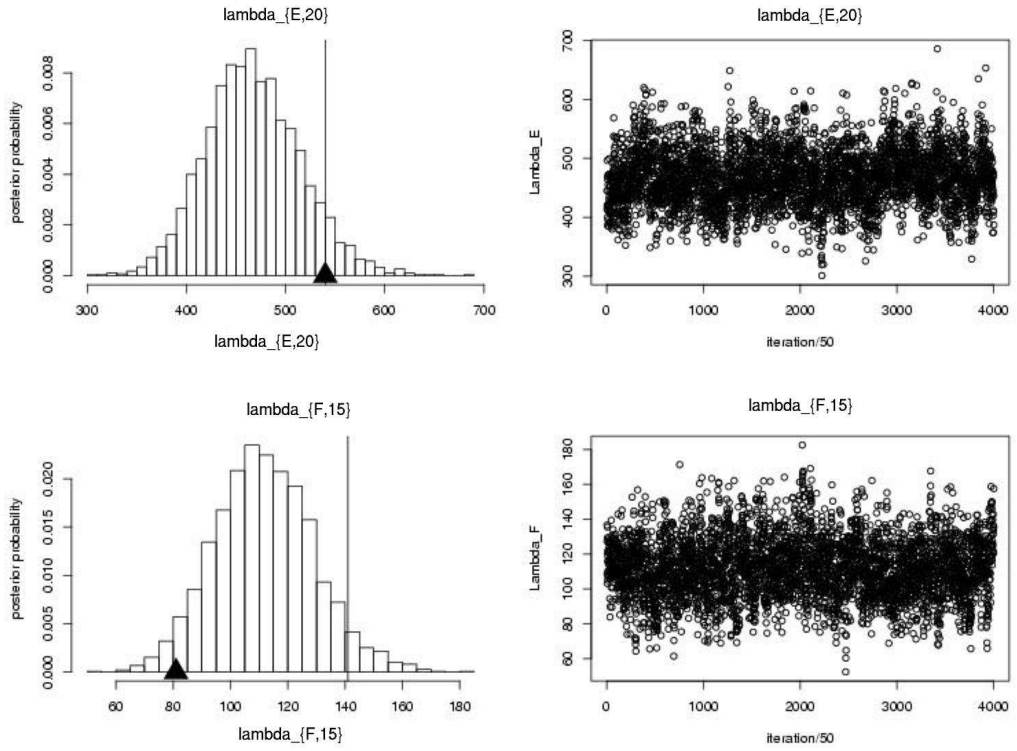

Figure 4: Histograms indicating the posterior distributions of two of the λ's, together with the λ's obtained by fitting to the actual data (vertical lines) and tuned data (triangles).

to manipulate a few parameters to try to correct the apparent major biases, than to jointly manipulate many parameters to obtain a more refined adjustment, as is done automatically in the Bayesian approach. (Of course, it is also important that the Bayesian approach indicates the very considerable uncertainty that exists in these parameters.)

Figure 5 shows the posterior distribution of the biases in the λ's at four of the input streets in the network. (The measurement error model was stated in terms of bias in counts, but that can directly be translated into bias in the λ's.) Here 1 (the vertical lines) would correspond to no bias. While most biases were positive, indicating (perhaps surprisingly) that the observers mostly over-counted vehicles, some were negative. Note also that the biases could easily be as high as 50%.

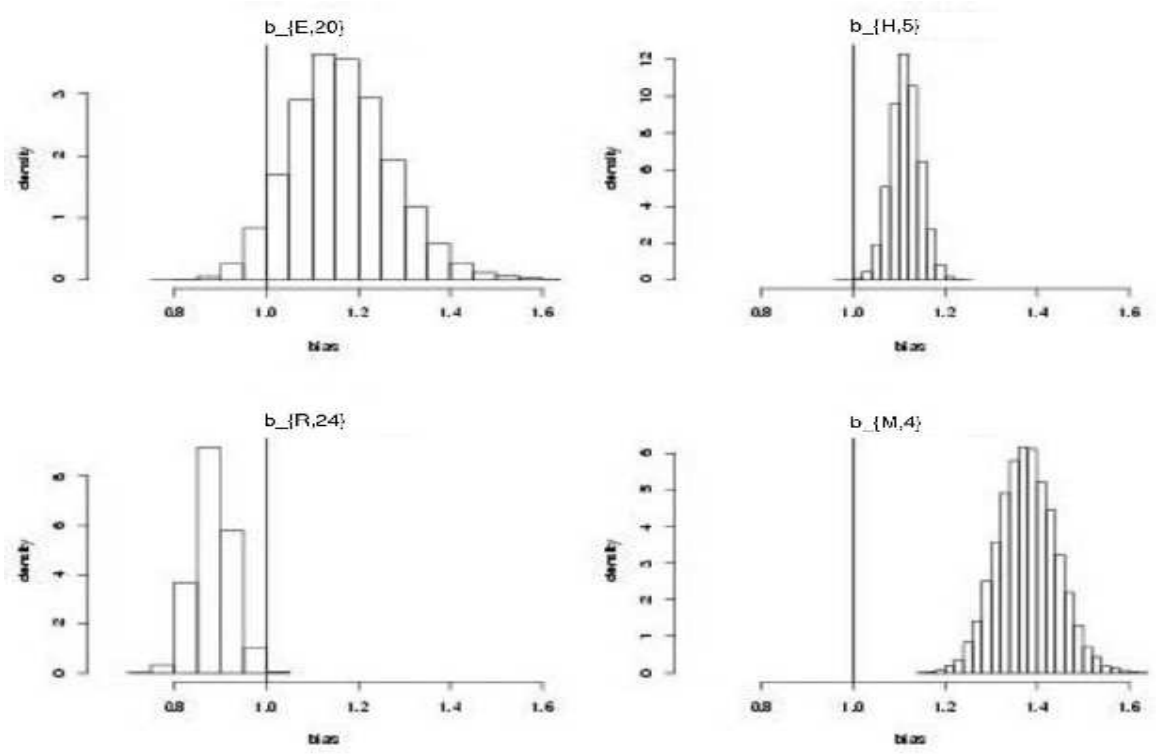

Figure 5: Posterior distributions of observer bias at four intersections.

Finally, the posterior distributions of two of the turning probabilities are given in Figure 6. Note that the distributions are quite disperse, partly justifying one of the approximations discussed in Subsection 2.2.

Acknowledgements

Research was supported by the U.S. National Science Foundation, Grant DMS-0073952.

Appendix: Details of the MCMC Analysis

FULL CONDITIONAL DISTRIBUTIONS:

The full conditional distributions of $\mathbf{P}$, $\mathbf{b}$, $\boldsymbol{\lambda}$ and β have closed form expressions. Specifically:

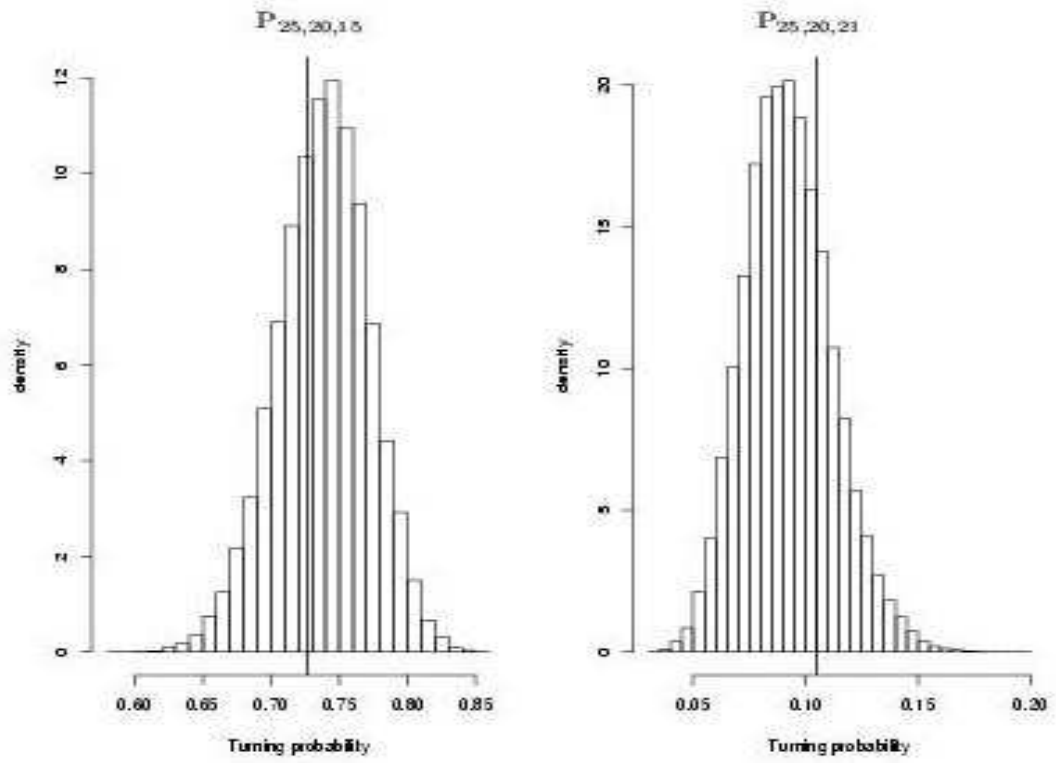

Figure 6: Posterior distributions of two typical turning probabilities.

Demand parameters. The λ_{ij}'s are (conditionally) independent with

$$\lambda_{ij} \mid \text{ rest } \sim \text{Ga}(N_{ij} + \frac{1}{2}, 1), \ i \in I_D \tag{A.1}$$

Turning probabilities. Let $\boldsymbol{P_{ij}}$ be the vector with components P_{ijk}, for $k \in D_{ij}$. In our net, the vectors $\boldsymbol{P_{ij}}$ have only two or three components, depending on the intersection. The components of $\boldsymbol{P_{ij}}$ obviously add to 1. Under independent Jeffrey's priors, all these vectors are (conditionally) independent. The full conditionals are of the form:

$$\boldsymbol{P_{ij}} \mid \text{ rest } \sim \text{Dir}(N_{ijL} + C_{ijL} + \frac{1}{2}, N_{ijT} + C_{ijT} + \frac{1}{2}, N_{ijR} + C_{ijR} + \frac{1}{2}), \tag{A.2}$$

where T,R and L refer to Through, Left and Right turns of cars travelling from i to j (D_{ij}). When only two-choice intersections are involved, these Dirichlets become betas. $\boldsymbol{P}$ is an 84-dimensional vector. Its joint full conditional distribution factors into the product of 32 Dirichlet and 20 beta distributions. Of course the N's in these expressions are the

values after the reparametrization discussed in Subsection 5.1.

Bias parameters. Each bias parameter has a (conditionally) independent full conditional of the form:

$$b_{ij} \mid \text{rest} \sim \text{Ga}(C_{ij\bullet} + \alpha, N_{ij\bullet} + \beta),\ i \in I_D.$$

Scale parameter. The full conditional of the scale parameter is simply a restricted Gamma. The full conditional for the shape parameter is log-concave, and thus is also easily sampled:

$$\begin{aligned} \beta \mid \text{rest} &\sim \text{Ga}(d\alpha + 1, \sum_{i \in I_D} b_{ij}) 1_{\beta > \frac{\alpha}{2}}, \\ \alpha \mid \text{rest} &\propto (\Gamma(\alpha))^{-d} \beta^{\alpha d} (\prod_{i \in I_D} b_{ij})^{\alpha} 1_{0<\alpha<2\beta}, \end{aligned}$$

where d=dim(I_D).

Latent vehicle numbers. The full conditional distributions of latent demand $N_{ij\bullet}$ and turning N_{ijk} true counts are discrete, and hence, in principle, easily sampled by the inverse cdf method. However, several difficulties are encountered.

After incorporating the restrictions on the N's in (4.13), the typical form of these full conditionals is a ratio of products of factorial terms in various of the N's with an additional multiplicative term of the form $h(p)^N$:

$$N_{ijk} \mid \text{rest} \sim \frac{\prod_s G_s}{\prod_t G_t} h(p)^{N_{ijk}}, \tag{A.3}$$

where s and t are the sets of all gamma functions G_s and G_t in the numerator and denominator, respectively, that, after reparametrization of the posterior, contain the latent count N_{ijk}. For example, in the restricted net RN, the full conditional of N_{I12} is

$$N_{I12} \mid \text{rest} \propto \frac{G_1\, G_2}{G_3\, G_4\, G_5\, G_6} \left(\frac{P_{I12} P_{123} (1 - P_{234})}{(1 - P_{I12})} \right)^{N_{I12}},$$

where $G_1 = \Gamma(N_{K2\bullet} + N_{I12} + N_{51\bullet} - N_{51J} - N_{\bullet 26} + 1)$, $G_2 = \Gamma(N_{I12} + N_{51\bullet} - N_{51J} + 1)$, $G_3 = \Gamma(N_{I1\bullet} - N_{I12} + 1)$, $G_4 = \Gamma(N_{I12} + 1)$, $G_5 = \Gamma(N_{K26} + N_{I12} + N_{51\bullet} - N_{51J} - N_{\bullet 26} + 1)$, and $G_6 = \Gamma(N_{K2\bullet} + N_{I12} + N_{51\bullet} - N_{51J} - N_{\bullet 26} - N_{234} + 1)$.

The full conditionals in (A.3) are only defined over the set of integers for which every argument of the gamma functions is nonnegative. (This was the big advantage of performing the reparameterization to eliminate the other restriction, leaving only nonnegativity re-

strictions.) Still, the domain over which (A.3) is defined depends on the values of the other N's and, hence, it is iteration-dependent. Solving the system of inequalities defined by the positivity of the arguments with brute force linear programming would be rather time consuming – especially for the full network, which has 90 N's (after reparametrization) – since it has to be done for each iteration. Luckily, since the reparameterization of the N's preserved the linearity of the argument in each gamma function, there is a systematic way to obtain this range. It is given by the following algorithm:

Step 1. For the argument in the gamma function G_i, determine whether N has a positive or negative sign and go to *Step 2* or *Step 3* accordingly.

Step 2. If N has a negative sign, define $S_i = \{0, \ldots, n_i\}$, where n_i is such that G_i evaluated at $N = n_i$ (all the other N's held constant) is 1 (i.e., the argument of the gamma function is 1).

Step 3. If N has positive sign, define $S_i = \{n_i, \ldots, \infty\}$, where n_i is defined as in *Step 2.*

Step 4. Once *Steps 1* to 3 have been done for all G_i in the full conditional, define the support of N as the intersection of all S_i

$$S = \bigcap_i S_i. \tag{A.4}$$

In the restricted net RN, this algorithm produces, for the example (N_{I12}) given before, $S_1 = \{\ (-N_{K2\bullet} - N_{51\bullet} + N_{51J} + N_{\bullet 26})\ ,\ \infty\ \}$, $S_2 = \{\ (-N_{51\bullet} - N_{51J})\ ,\ \infty\ \}$, $S_3 = \{\ 0\ ,\ N_{I1\bullet}\ \}$, $S_4 = \{\ 0\ , \infty\ \}$, $S_5 = \{\ (-N_{K26} - N_{51\bullet} + N_{51J} + N_{\bullet 26})\ ,\ \infty\ \}$, and $S_6 = \{\ (-N_{K2\bullet} - N_{51\bullet} + N_{51J} + N_{\bullet 26} + N_{234})\ ,\ \infty\ \}$, and the range of N_{I12} in a specific iteration would be

$$S = \bigcap_{i=1}^{6} S_i = \{max(0, N_{\bullet 26} - N_{K26} + N_{51J} - N_{51\bullet}, N_{234} + N_{\bullet 26} + N_{51J} - N_{51\bullet} - N_{K2\bullet})\ ,\ N_{I1\bullet}\ \}$$

This algorithm always produces the required range because the linearity of the restrictions in N, after the reparametrization (5.16), guarantees a linear functional form of the arguments of the gamma functions (only one intercept with the x-axis for these arguments). It is much more efficient than brute force linear programming, because it only requires a single evaluation per iteration of the sampler and per parameter N. Moreover, if the support in the previous iteration was non-empty, then the support in the current iteration will be non-empty.

STARTING VALUES FOR THE MCMC:

For most of the parameters, choice of the starting values presents no problem. MLE's for the $\boldsymbol{\lambda}$ and P, $b_{ij} = 1$, and $\alpha = \beta = 0.8$ provide good initial choices.

The difficult initialization is the choice of the N's, since one must have a compatible choice of all 90 parameters, i.e., a choice which is in the intersection of all S_i. This appears to require finding a solution to a system of restricted diophantine linear equations, a problem that has an extensive literature in mathematics and computer science. Our case is simpler than the general case, however, in that the observed input counts are also unknown, so we can choose their starting values high enough to ensure a solution. Here is the algorithm we use to obtain a starting value.

For all N's *before* reparametrization:

Step 1. If an output video count is taken between j and k ($k \in I_V$), assign to N_{ijk} and $N_{ij\bullet}$ (for $i \in I_D$) the observed video count ($C_{\bullet jk}$).

Step 2. If an input video count is taken between j and k ($j \in I_V$), assign to N_{ijk} the observed video count $C_{\bullet jk}$.

Step 3. Transmit the values of the remaining observed inputs to any non-video exit.

Step 4. Set any remaining N's to zero.

References

[1] Chen, M.H., Shao, Q.M. and Ibrahim, J.G. (2000), *Monte Carlo Methods in Bayesian Computation*, New York: Springer-Verlag.

[2] Cowell, R.G., Dawid, A.P., Lauritzen, S.L. and Spiegelhalter, D.J. (1999), *Probabilistic Networks and Expert Systems*, New York: Springer-Verlag.

[3] Robert, C.P. and Casella, G. (1999), *Monte Carlo Statistical Methods*, New York: Springer-Verlag.

[4] Sacks, J., Rouphail, N.M., Park, B., and Thakuriah, P. (2000), "Statistically-based validation of computer simulation models in traffic operations and management," *Technical Report* 112, National Institute of Statistical Sciences (US).

[5] US Federal Highway Administration (1997), "CORSIM User's Manual", FHWA, US. Department of Transportation Office of Safety and Traffic Operation R&D, Intelligent Systems and Technology Division, McLean, VA.