School of Engineering

MSc(Eng) Project

Day-to-day tra\_c demand and ow

dynamics with autonomous vehicles

William Kane

Student No: 201068461

Risk and Uncertainty

Supervisor: Hongbo Ye

Project: 200

School of Engineering

University of Liverpool

Brownlow Hill, Liverpool, L69 3GH

August 2020

Project Report William Kane

Summary

To aid the planning of roads and infrastructure it is important to be able to model tra\_c

patterns. When presented with a road network, drivers must make a decision about

which path to take, and the travel time they experience on a given road is function of

the amount of tra\_c, the road's capacity and the travel time on that road in the absence

of tra\_c. E\_ective modelling of the drivers' decision making can be used to optimise

road network design with the goal of minimising travel times. Early methods used models

based on game theory, this led to more detailed day-to-day deterministic models being

formulated, and then stochastic models which model the probabilities of a given path being

chosen. The twenty-\_rst century has seen agent-based modelling and simulation become

the preferred method. Drivers are modelled separately as heterogeneous agents with

independent knowledge and decision making, and the simulations are studied to observe

the emergent behaviour of the system. The twenty-\_rst century has also seen growing

interest in driver-less autonomous vehicles with computer-controlled decision making, and

access to more information than a human driver through network-wide systems. It is

believed that autonomous vehicles will increase the capacity on road networks and shorten

travel times as they can react faster and access better information. This report develops

an agent-based model and simulation for studying day-to-day demand and ow dynamics

on a road network with mixed human and autonomous drivers. The model is analysed,

and key features are compared to existing literature. The analysis shows overly rational

autonomous vehicles do not improve tra\_c ow when mixed with human drivers, and

irrationality is key to maintaining fast travel times.

i

Project Report William Kane

Contents

Summary i

Table of Notation iv

1 Background 1

1.1 Traditional Models of Network Flow . . . . . . . . . . . . . . . . . . . . . . 1

1.2 Traditional Models of Tra\_c Flow . . . . . . . . . . . . . . . . . . . . . . . 1

1.3 Modelling Humans . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 2

1.4 Agent-based Modelling and Simulation . . . . . . . . . . . . . . . . . . . . 2

1.5 ABMS for Travel Behaviour . . . . . . . . . . . . . . . . . . . . . . . . . . 3

1.6 Autonomous Vehicles . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 3

2 Aims and Objectives 5

2.1 Aims . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5

2.2 Objectives . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5

3 Literature Review 6

3.1 Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 6

3.2 Classical Literature . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 6

3.3 ABMS for Tra\_c Flow . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 7

3.4 Agent Rationality and Memory . . . . . . . . . . . . . . . . . . . . . . . . 10

3.5 Advanced Traveller Information System . . . . . . . . . . . . . . . . . . . . 13

3.6 Route Choice Rules . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15

3.7 Autonomous Vehicles . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 16

3.8 Conclusion and Research Opportunities . . . . . . . . . . . . . . . . . . . . 18

4 Methodology/ The Model 20

4.1 Overview . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 20

4.2 Program . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 22

4.3 Usage . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 23

5 Results 25

5.1 Default Simulation . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 25

5.2 E\_ects of AV Proportion . . . . . . . . . . . . . . . . . . . . . . . . . . . . 25

5.3 E\_ects of \_hv . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 25

5.4 E\_ects of \_av . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 26

5.5 E\_ects of Lhv . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 29

5.6 E\_ects of Lav . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 30

5.7 E\_ects of \_ . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 30

ii

Project Report William Kane

6 Discussion 33

6.1 Overall Behaviour . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 33

6.2 AV Proportion . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 33

6.3 Rationality . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 34

6.4 Memory . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 35

6.5 Comparison with literature . . . . . . . . . . . . . . . . . . . . . . . . . . . 36

7 Conclusions 37

Bibliography 38

Appendix 1: Python Code 40

Appendix 2: Gantt Chart 48

iii

Project Report William Kane

Table of Notation

[[[ TODO ]]]

iv

Project Report William Kane

1 Background

1.1 Traditional Models of Network Flow

In mathematics and graph theory (Jungnickel 2013), a ow network is a directed graph, or

digraph, with limited capacity per edge. A digraph is a set of vertices v 2 V connected by

edges e 2 E, where each edge has a start vertex and an end vertex. A ow is a map f(e)

from each edge to any non-negative real number such that the following two conditions

are met:

• 0 \_ f(e) \_ c(e) for all edges, where c(e) is the capacity on edge e.

•

P

e+=v f(e) =

P

e􀀀=v f(e), where v is not a source or sink. i.e. the total ow into

vertex v is the total ow out.

Methods such as the Ford-Fulkerson Algorithm can be used to calculate the maximal

possible ow between sources and sinks of the digraph (Ford & Fulkerson 1956). Created

by Lester R. Ford, Jr. and Delbert R. Fulkerson in 1953, the algorithm was used to

optimise ow on rail networks between two locations. Faster algorithms have since been

discovered.

1.2 Traditional Models of Tra\_c Flow

In the study of tra\_c ow on road networks, digraphs are usually called networks, vertices

are called nodes, and edges are called links. In addition to capacity, each link has a travel

time. When working with travel ow, the capacity constraint is relaxed so that the ow on

a link can be over capacity, at the cost of higher travel times. Rather than the ow being

designed as with a rail network, the ow is a result of the routes chosen by drivers as they

drive from their origin to their destination. The problem is to model this route choice

behaviour to aid the design of road networks, with the goal of minimising travel times

for all drivers, and is known as the tra\_c assignment problem. Here traditional models

refers to approaching the problem with game theory, deterministic models, or stochastic

models.

John Wardrop was the \_rst to study this problem in 1952 (Wardrop 1952) by applying

game theory to tra\_c ow. He found an equilibrium point based on the Nash Equilibrium

now known as the Wardrop user equilibrium. Later research on route choice modelling

took two approaches, deterministic and stochastic.

A deterministic user equilibrium (DUE) can be found be if travel times on a link are

expressed as a function of the ow on the link, creating a mathematical optimisation

problem of \_nding the minimal feasible travel times over all links. DUE methods assume

drivers all have complete information on the network and always chose the route with the

1

Project Report William Kane

lowest travel time. To improve on this a stochastic user equilibrium (SUE) introduces

errors, where drivers only have their perceived travel times.

Daganzo & She\_ (1977) suggested a SUE model with each drivers’ route choice probabilities

being a function of their perceived travel times but found these random e\_ects still

lead to a \_xed equilibrium with a large number of drivers.

An improvement on user equilibrium models is day-to-day models. These introduce time

as a parameter in the model and do not assume equilibrium forms. These models are

solved either analytically or through simulation. Simulation provides the most realistic

way to model human decision making.

1.3 Modelling Humans

Human behaviour is complex and di\_cult to model. Lee et al. (2010) classi\_ed di\_erent

attempts into economics, psychology, and synthetic engineering based approaches. An

economic approach is based on the assumption decision makers are rational, however this

ignores the human nature of decision making such as incomplete information, memory,

stress and fatigue (Zheng et al. 2013). To address this, psychologists attempt to model

human behaviour but often do so in a less quantitative way and are based on laboratory

experiments, these controlled environments present people with static decision making

problems that often don't compare well to the real world. The synthetic engineering

approach can incorporate both economic and psychological elements and attempts to

reverse engineer human behaviour in real environments.

Among synthetic engineering approaches Lee. singles out Soar, Adaptive Control of

Thought Rational (ACT-R), and BDI. Soar and ACT-R are complex models that attempt

to produce a universal theory of cognition by considering the actual mechanisms

of the brain, but are di\_cult to understand or deploy. However, BDI is simple enough

to be easily written in a programming language, for this reason it is often deployed in

successfully in systems such as air tra\_c control. It is this framework of beliefs, desires,

and intentions that underlay how humans will be modelled as agents.

1.4 Agent-based Modelling and Simulation

Agent-based modelling and simulation (ABMS) breaks a system into discrete interacting

units. These are allowed to interact with each other and an environment, based on simple

rules, and the emergent system behaviour is observed. Early examples of ABMS include

John Conway's Game of Life (Gardner 1970), in which a grid of cells, either alive or dead,

would live or die depending on the state of the neighbouring cells. Through simple rules

it is possible to create complex emergent behaviour, and people have even created basic

computers within the Game of Life. However, ABMS is computationally expensive and

2

Project Report William Kane

often impossible to reproduce, but modern computer power makes it a practical modelling

tool for complex systems.

Macal & North (2005) de\_ned the properties of an agent as being:

• Discrete, self-contained units with clear boundaries.

• Situated in an environment which has ruled of interaction. Agents can respond to

their environment and each other.

• Goal-orientated such as having an objective and cost function.

• Autonomous and able to function independently without other agents.

• Flexible and able to learn from experiences.

ABMS can be built on rational economic or psychological assumptions and allows for a

complex model to be broken down into simple components. Each agent can be modelled

under the BDI framework, with belief in the form of memory, desires in the form of goals

and cost functions, and intentions in the form on interactions they decide to make. By

running simulations, it is possible to observe emerging patterns and behaviour in a natural

and exible way (Bonabeau 2002).

1.5 ABMS for Travel Behaviour

Traditional methods of modelling tra\_c ow treat all drivers at homogeneous. ABMS

allows for each driver to be modelled independently with separate memory, goals, and

decisions. It also allows for limited rationality and varying decision-making processes. It

is natural when facing the tra\_c assignment problem to give each agent-driver knowledge

of the available routes and expected travel times, the desire to get from their origin to their destination with minimal travel time, and the intention to choose a route. This approach

is more realistic than traditional methods which look at the system top-down and try to

model the entire system's behaviour. The bottom-up approach of ABMS is exible and

can be applied to hypothetical networks by focusing on why the system behaves as it does

(Zheng et al. 2013). The advantages of ABMS make it the preferred method to modelling

tra\_c ow.

1.6 Autonomous Vehicles

Development of autonomous vehicles is advancing fast since Google began leading the

industry in 2009. They are predicted to become available to consumers in the 2020s,

onto road networks originally designed for human-driven vehicles (Bertoncello & Wee

2015). Ackerman (2012) predicts AVs will signi\_cantly increase road capacity, this e\_ect

3

Project Report William Kane

is already being seen with more advanced adaptive cruise control systems on current

vehicles (Ntousakis et al. 2015).

4

Project Report William Kane

2 Aims and Objectives

2.1 Aims

The aim of this project is to investigate the e\_ects of autonomous vehicles in a mixed

road network with both human and autonomous vehicles, using agent-based modelling

and simulation.

2.2 Objectives

The main objectives of this project are as follows.

• To develop a program to simulate drivers day-to-day route choice behaviour.

• To analyse the impact of di\_erent driver behaviours and system parameters.

• To analyse the impact of di\_erent model assumptions deemed relevant to the di\_erence

in behaviour between autonomous and human drivers.

• To develop a model to simulate a mixed system of autonomous and human drivers,

and their route choice behaviour.

• To determine what behaviours and parameters for autonomous drivers optimise the

ow through the network.

5

Project Report William Kane

3 Literature Review

3.1 Introduction

Relevant literature on modelling tra\_c ow through a network dates from the 1950s to

the present. Academic study of the tra\_c assignment problem began in the 1950s with

the work of John Wardrop and his game theory based user equilibrium models. Wardrop

notes before this theoretical approach, study of tra\_c was based on 'before and after'

methods looking at the e\_ects of a change to road design, and learning from the change

to inuence future planning.

Literature following Wardrop continued with user equilibrium, using deterministic and

stochastic models to \_nd the tra\_c ows at their equilibrium. Stochastic user equilibrium

allow for random errors in perceived travel times, such as in Daganzo & She\_ (1977).

These equilibria may be unstable as was found by Horowitz (1984). This leads to the

need for time-dependant dynamical models such as found in Hazelton & Parry (2016).

In the twenty \_rst century, it has become common to approve the tra\_c assignment

problem with agent based models and simulations. Zheng et al. (2013) provide a detailed

primer on AMBS for tra\_c ow. They argue its uses and derive a basic formulation that

leads to e\_ective modelling that agrees with classical models.

Nakayama et al. (2001) explores using classical models the ways in which drivers choose

a method or choose a route, \_nding humans do not tend to become rational. Its in

irrationality of humans that is di\_erent to how computers do decision making.

Autonomous vehicles have the potential to be more rational than humans. However the

work of Liu & Huang (2007) shows that rationality and perfect information may do more

harm than good and lead to slower travel times over all. It is natural to wonder if

autonomous vehicles will be too rational.

No matter how rational autonomous vehicles are, or how rational humans are, computers

can react faster than humans. Faster reaction times will lead to safer driving. In the

paper by Chen et al. (2017) the e\_ects of grouping and spacing of AVs is explored, noting

how more AVs will lead to high capacities on roads, thus faster travel times.

3.2 Classical Literature

Wardrop (1952) introduced a theoretical approach to predicting tra\_c ow known as the

User Equilibrium. Wardrop notes before this theoretical approach, study of tra\_c was

based on 'before and after' methods looking at the e\_ects of a change to road design,

and learning from the change to inuence future planning. and engineers learned from

experience. His user equilibrium occurs when no driver can choose a better route, a form

6

Project Report William Kane

of Nash Equilibrium, where any change will increase their travel time. This two criteria

for a user equilibrium are, quoting his 1952 paper:

1. The journey times on all the routes actually used are equal, and less than those

which would be experienced by a single vehicle on any unused route.

2. The average journey time is a minimum.

This is the earliest form of Deterministic User Equilibrium (DUE), which assumes all

drivers are rational, homogeneous and have access to perfect information, and solves for a

equilibrium state deterministically. There is no e\_ects of past route choices on future route

choices in DUE (Zhang 2011). The simplicity of DUE models means they are still in use

today. Many methods of solving for the equilibrium state exist such as the Frank-Wolfe

algorithm (Frank et al. 1956) or the OBA algorithm

DUE assumes all drivers are rational, homogeneous and have access to perfect information.

These are not natural assumptions. To address this stochastic user equilibrium models

allow for errors in perceived travel times, and random choice of a route, dictated by a

probably distortion. Daganzo & She\_ (1977) were the \_rst to generalise the DUE to

account for uncertainty in travel times. Under this new probabilistic Stochastic User

Equilibrium (SUE) each driver only believes they cannot improve their travel time. Each

driver has a random error on their perceived travel times, usually modelled with a normal

or Gumbel distribution. Many algorithms exist to \_nd the SUE, some are detailed in

Prashker & Bekhor (2004).

Horowitz (1984) looked into the stability of SUE and found the conditions for stability

were quite restrictive. As such the idea that tra\_c reaches an equilibrium cannot be

trusted leading to the need for more dynamic models.

By the 1980s there was growing interest in models that could provide real-time information

and equilibrium models. This led to within-day models and day-to-day models Watling &

Hazelton (2003). Day-to-Day (DTD) models are discrete time and can be deterministic

or stochastic, and are solved analytically or by simulation (Peeta & Yang 2003). The

next section will focus on simulation models and analytical models are considered here as

classical models.Analytical methods of stochastic DTD models include Cascetta (1989)

who uses a Markov model to \_nd a steady state. Hazelton & Parry (2016) provide a

comparison of various methods. Li et al. (2018) used a DTD model to look at mixed human

vehicle and autonomous vehicle tra\_c and found sluggish convergence to equilibrium. In

particularly the found a higher proportion of AVS slowed converge down.

3.3 ABMS for Tra\_c Flow

Zheng et al. (2013) breaks down the bene\_ts of ABMS for tra\_c ow. They give these

7

Project Report William Kane

bene\_ts as:

• ABMS captures individual's rational nad irrational behaviour that are di\_cult to

quantify.

• AMBS treats drivers as heterogeneous, with di\_erent habits and preferences.

• It captures the e\_ects of heterogeneity and emergent collective behaviour.

• It allows all drivers to have di\_erent information.

• It is possible to run real-time simulations, rather than day-to-day.

• It allows for simpler formulation of complex decision making.

• It can test drivers with sudden change in the network.

• You can observe emergent behaviors after a change is made.

Zheng et al. then go on to derive a formulation for ABMS for tra\_c ow, with each

driver being an agent. The drivers start with little information, with initial perceived

travel times initialised from their expectation, which can be the free ow travel times.

TTn

min is the perceived shortest time on day n and route j, and \_ is a small perception

error. From this each driver has 3 rules for day n + 1:

• If (TTn

j = TTn

min) then they stick to the same route.

• If (TTn

j 􀀀 TTn

min) \_ \_ then they stick to the same route.

• If (TTn

j 􀀀TTn

min) \_ \_ then they believe another may be faster and consider changing,

changing with probably of (TTn

j 􀀀 TTn

min)=TTn

j

They next formulate how a driver chooses a route, when the decide to change route.

They base this around a Bayesian learning process. They assume the probabilities of

each route prior to learning is f(Pn), given by a Dirichlet distribution with parameter

\_n = (\_1n; \_2n; : : : ; \_jn) where Pn is the vector of probabilities for choosing each route.

They let dn be a vector where dj;n = 1 if route j is perceived to be the shortest route,

else 0. They assume the subjective probability

g (dn j Pn) =

Y

j2J

pdjn

jn :

They note the mean of each random variable in the Dirichlet probability vector is given

by

E(Pj;n) = aj;n=a0n

where a0;n is the sum of aj;n over all j. This leads to the updated posterior probabilities

for day n + 1, where route i was taken on day n:

pi;n+1 = (\_in + 1) = (\_0n + 1) i 2 J

8

Project Report William Kane

pj;n+1 = \_jn= (\_0n + 1) ; j 2 J; j 6= i

So the Gumbel probability distribution for the probability of each route on day n + 1 is

given by equation 1 with parameter \_.

pj;n+1 =

e\_(\_jn+djn)

P

j02J e\_(\_j0n+dj0n)

(1)

Using this model, they ran a simple simulation on a network of two nodes O and D and

3 links each from O to D, with capacities 200, 400, and 300, and free-ow travel times of

10, 20, and 25 respectively. They ran 1000 agents driving from O to D, with a1 = (1; 1; 1)

giving each route an even starting chance.

They calculated the travel times using the Bureau of Public Roads (BPR) function, which

is common in most literature (Manual 1964) , and given by equation 2. cl represents the

travel time on link l, c0l

is the free-ow travel time, vl is the tra\_c on the link, and Yl is

the capacity of the link.

cl = c0l

(1 + 0:15(vl=Yl)4) (2)

They found the travels times converged after about 500 iterations to the time predicted

from classical models, namely the Frank{Wolfe Algorithm using the convex combination

method, with times on all routes about 25:4s. There results are shown in \_gure 1 taken

from their paper.

Figure 1: Results of simulation from Zheng et al. (2013)

9

Project Report William Kane

3.4 Agent Rationality and Memory

Wei et al. (2014) used AMBS to explore the e\_ects rationality and memory to route choice

behaviour.

They set out a model using reinforcement learning. With reinforcement learning, actions

which lead to better outcomes are more likely to be repeated in the future, compared to

actions that lead to worse outcomes. The reinforcement learning model they use is the

Bush Mosteller (BM) model from literature (Cross 1973). Under this model each action c

has a stimulus Sic

on day i given by equation 3. uc is the payo\_ for action c and Ai desired

payo\_. The denominator represented the highest value of the uc 􀀀 Ai over all actions c.

Once the stimulus for each action is found, the probabilities of each action are updated

by equation 4, where l is the learning rate. A higher l allows the player to learn faster.

St

c =

uc 􀀀 Ai

sup ju(k) 􀀀 Aij

(3)

pt+1

c =

\_

pt

c + (1 􀀀 pt

c) lst

c; if st

c \_ 0

pt

c + pt

clst

c; if st

c < 0

(4)

They apply this to the tra\_c assignment problem using ABMS where each agent's actions

are the di\_erent route choices available, using travel times as payo\_. For the desired payo\_,

they use a weighted average of all the experienced travel times given by

Ati

=

Pt􀀀1

j=1 t􀀀1􀀀jTj

i Pt􀀀1

j=1 t􀀀1􀀀j

for agent i on day t. Tj

i is the travel time on day j and is the weighing parameter are

represents the memory level. For the pay o\_ of a given route c they use

Mt

ik =

Pt

j=1 t􀀀jTj

i \_j

ik Pt

j=1 t􀀀j\_j

ik

Mt

ic =

XN

k=1

Mt

ik\_t

ik

where \_ = 1 if path k was taken on day j, else \_ = 0.

By using these and 3 and 4 each agents can calculate the probabilities of choosing each

route as Pi = fpi

1; pi

2:::g. A route is then selected when each agent generates a random

number b and chooses route c by equation 5

Xc􀀀1

j=1

pt

j < b \_

Xc

j=1

pt

j (5)

10

Project Report William Kane

They used this ABMS model on three test networks. Travel times are calculated with the

BPR function in equation 2. They explored the e\_ects of the two key parameters: the

learning rate l, and the memory level .

To explore they \_xed l = 0:3 and tried = 0:1; 0:5; 0:9. The results are shown in \_gure

2 lifted from their paper. For lower memory levels the ows fail to converge to the DUE,

but instead uctuate around it. The standard deviation of these ows decreases are

increase. They note this makes sense as agents with better memory are able to make a

better judgement.

Figure 2: E\_ects of memory on stability from Wei et al. (2014)

To explore l they then \_xed = 0:6 and tried l = 0:1; 0:5; 0:9. These results are shown

in \_gure 3 lifted from their paper. They only consider l = 0:1 to have converged while

the other two did not. They note in the l = 0:9 case the ow converges to the DUE then

leaves it again, and repeats that in a cycle. In the l = :5 case the ow uctuates around

di\_erent values to the DUE. They explain this behaviors by noting that a higher l leads

11

Project Report William Kane

to faster change in the probabilities, leading to more variation. Lastly they note a higher

\_xed slightly reduces the variation seen.

Figure 3: E\_ects of learning rate on stability from Wei et al. (2014)

From this paper it can be seen that the intuitive idea that better memory and faster

learning may not lead to stability. Irrationality may be import to stabilising ow on a

network.

12

Project Report William Kane

3.5 Advanced Traveller Information System

Liu & Huang (2007) built an agent-based model on a small grid network, with an Advanced

Traveller Information System (ATIS). The network had 9 nodes 12 links, and is displayed

later in this report as Test Network 1, in \_gure 7 with capacities and free ow in tables

4. The capacities mentioned in 4 are double those use by Liu & Huang. Travel times are

calculated with the BPR function again, given before in equation 2.

They \_rst solved a stochastic user equilibrium model, with the probabilities of each route

given be a Gumbel distribution identical equation 1. They note on the parameter \_

\Higher is [\_]-value, wiser drivers make route choice."" A higher \_ value makes the fastest

route more probable. They solve a convex minimisation problem on 1 to \_nd the equilibrium

point. The results with di\_erent \_ are shown in table 1 which is lifted from

their paper, note in this table \_ = . The results show the ow along route (1; 4; 7; 8; 9)

converging to a value as the drivers become more rational, especially when \_ 0:5

Table 1: SUE results taken from Liu & Huang (2007)

Next they set up an ABMS model with an Advanced Traveller Information System. The

ATIS allows drivers to look up the exact travel times, without perception error, on all

routes from the previous day. Each agent weighs up this information with their perceived

travel time on each route. Their new perceived travel times are given by equation 6 where

agent i on day t + 1, for route r and OD w, can look up the exact travel times c(i;t)

rw and

mix them with their perceived travel times \_ (i;t)

rw with parameter \_.

When ATIS information is unreleased agents can only update perceived travel times on

routes they have taken. They are able to weigh in day n travel time c(i;t)

kw with there

expected travel time for the route k 2 Rk they chose, else they cannot update the travel

time. This is shown in equation 7. In this case parameter \_ is known as the learning rate.

\_ (i;t+1)

r;w = \_c(i;t)

rw + (1 􀀀 \_)\_ (i;t)

rw ; \_ 2 (0; 1] (6)

(

^\_ (i;t+1)

kw = \_c(i;t)

kw + (1 􀀀 \_)^\_ (i;t)

kw ; k 2 Rw; \_ 2 (0; 1]

\_ (i;t+1)

rw = \_ (i;t)

rw ; r 2 Rw; r 6= k

(7)

They ran a 2 simulation with 500 agents with OD node 1 to node 9. One simulation used

an ATIS and one did not. The results are shown in table 2 which is lifted from their

13

Project Report William Kane

paper. It can be seen that in both cases the ow converges to the equilibrium value from

the SUE model, but it converges faster under the ATIS. However this behaviour breaks

down as agents become too trusting of the ATIS. In \_gure 4 lifted from their paper it can

be seen that standard deviation of the ow along route (1; 4; 7; 8; 9) increases with \_, and

increases sharply for \_ \_ 0:25. Too much trust in the ATIS leads to instability.

Table 2: Simulation results taken from Liu & Huang (2007)

Figure 4: Simulation results taken from Liu & Huang (2007)

Their simulation results drew two key conclusions. Firstly, the system converged to the

equilibrium predicted from a stochastic model faster when the ATIS system was used.

Secondly, if the drivers put too much trust in that information the equilibrium becomes

unstable.

14

Project Report William Kane

3.6 Route Choice Rules

Nakayama et al. (2001) used ABMS to study the e\_ects of heterogeneous route choice

rules among di\_erent drivers. The 4 rules they explored were:

1. No switching

2. Random switching

3. Experience-only based switching

4. Full rational switching

Rules 1 and 2 are simple. Under rule 1 a driver never changes route after their initial

selection and under rule 2 drivers choose a route without thinking, in a uniformly random

manor.

A driver using rule 3 only chooses a route based on experienced travel times according

to equation 8. Here ECk

i;j is the expected cost of route j on day i, under rule k,

tavg

􀀀

tij ; nki

\_

returns the average experience travel times over the last n days, from set tij ,

and k

i

\_

tmax (tij) 􀀀 tmin (tij)

captures the variability of travel times on route j, with k

i

being the drivers attitude to uncertainty. k is included to label di\_erent formulations of

this rule, each labelled with di\_erent k, having di\_erent n and .

ECk

i;j = tavg 􀀀

tij ; nki

\_

+ k

i

\_

tmax (tij) 􀀀 tmin (tij)

(8)

A driver using rule 4 simply uses the average of all the travel times they have experienced

on each route, choosing the smallest. Unlike in rule 3 where only the past n days are

considered, a rational rule 4 driver remembers all travel times.

Next they allow each agent in their simulation to change rule dynamically based on which

rule is working best for them at that moment. Each rule l has a performance indicator f

given by equation 9, where ti is the travel time the agent experienced on day i and s is a

parameters reecting how much they learn from experience. Each driver choose the rule

l with minimal fl on day i + 1.

fl

i+1 =

\_

(1 􀀀 s)fl

t + sti if route used

fi otherwise

(9)

The study ran an agent-based simulation on 4000 drivers on a network with 2 routes

between the agents OD. Travel times were again calculated with the BPR in equation 2,

but the 0:15 parameters and 4 parameter are instead both 2. Route 1 has capacity 4000

and free ow travel time 20, while route 2 has capacity 2000, and free ow 10. They

solved

15

Project Report William Kane

For their analysis, parameter s = 0:9. Figure 5 shows the number of drivers that used

each route choice rule, lifted from their paper. It can be seen that drivers did not converge

to any rule and all four rules remained in use and somewhat stable. However travel times

on all routes did stabilise to the times predicted using a deterministic user equilibrium.

Figure 5: Simulation results taken from Nakayama et al. (2001)

The author was unable to \_nd a colour copy of this \_gure with all four lines clearly

visible.

The \_nding of their paper do not \_t with the assumption made by top-down models that

all drivers are rational and homogeneous as all four rules remained in use through out

the simulation, showing agents remained irrational with no incentive to learn to become

rational. This contraction goes against classical models of network ow, and highlights

the need for ABMS.

3.7 Autonomous Vehicles

The challenge with route choice for autonomous vehicles is predicting interaction between

human driven vehicles and AVs CITE(Li, Liu and Nie, 2018).

Chen et al. (2017) looked into the road capacity on networks with both human and

autonomous vehicles. They set out to build a systematic formulation of the operational

capacity of roads and the e\_ects of di\_erent AV penetrations. While there paper explores

in detail the e\_ects of dedicated AV lanes on roads, the content highlighted will focus on

single-line mixed tra\_c, with AVs groups of tra\_c called platoons of length n.

16

Project Report William Kane

Figure 6: Spacing of HVs and AVs from Chen et al. (2017)

They explore spacing between vehicles, based on the \_eld test of CITE [Milan\_es et al.,

2014; Shladover et al., 2010] and identify four spacing levels. The standard spacing is S0,

this represents the spacing, front bumper to front bumper, of regular HV tra\_c. They

assume a HV will maintain at least the regular spacing behind an AV, so the spacing

between an AV in front to a HV behind is \_Rs0 where \_R \_ 1. AVs travelling in a

platoon of length n will group closer together than normal HV tra\_c, as they are assumed

to have faster reaction times thus needing a smaller safe distance to the vehicle ahead.

As as results the spacing between AVs in a platoon is s0 where < 1. Lastly an AVs

travelling behind a HV, or a di\_erent AV platoon will leave more space than is found

within their platoon, so the spacing is \_As0 where \_A \_ . All these spacing are \_xed and

deterministic, and should be through of as long-term averages. Figure 6 lifted from their

paper shows all 4 levels of spacing, where m is the average length of HC tra\_c. \_R; and

\_A \_ 0

From these spacings, they \_nd the average spacing as

\_ S =

\_AS0 + (n 􀀀 1)S0 + ~m

􀀀

\_RS0 + (m 􀀀 1)S0

\_

n + m

where ~m is 1 when m > 0 and 0 when m = 0, to indicate whether HVs are present. Even

when m = 0 AVs will not form into one continuous platoon as it is assumed platooning

requires communications between vehicles, which is only possible over a \_nite range. From

this they de\_ne the av gain " as given by equation 10 where \_ is the proportion of AVs

to the total tra\_c.

" =

(

1 􀀀 􀀀

\_

\_A􀀀

n + \_R􀀀1

n

\_

; if 0 \_ \_ < 1

1 􀀀 􀀀 \_A􀀀

n ; if \_ = 1

(10)

Note as \_R \_ 1; < 1 and \_A \_ , " < 1. Using equation 10 the average spacing can be

rewritten as

\_ S = S0(1 􀀀 \_")

17

Project Report William Kane

so it is clear that a higher \_ leads to smaller average spacings. Chen et al. then use this

to derive the capacity as a function of \_ and " stating that, when C0 is the capacity of a

road under just HV tra\_c, with AVs and HVs the capacity is adjusted by equation 11.

C = f(\_; ") =

C0

1 􀀀 \_"

(11)

From equation 11 they derive four intuitive rules:

• Capacity increases with platoon size n

• Capacity decreases with

• Capacity decreases with \_A

• Capacity increases with AV proportion \_ at an increasing rate.

From this work it is clear and AMBS using mixed HV/AV tra\_c would need to consider

link capacity as a function of the proportion of AVs in the tra\_c, and the AV gain, de\_ned

by setting values for the spacing parameters.

3.8 Conclusion and Research Opportunities

Methods exist to tackle the tra\_c assignment problem, from the early DUEs of Wardrop

to the SUEs of Daganzo and She\_ and Cascetta. However these homogeneous models fail

in that they assume drivers to be rational, which isn't the case as is shown by the work of

Nakayama et al. (2001) where driver-agents did not become rational when learning how

to choose a route. It is for this reason that agent based modelling and simulation has

emerged as the contemporary approach to modelling tra\_c ow, and this method will be

used in this report. The case for ABMS is made well in the primer by Zheng et al. (2013).

The work of Liu & Huang (2007) raises an import concern when thinking about autonomous

vehicles. Too much trust in an advanced traveller information system, through

which drivers can look up past travel times on any route, leads to instability. It is reasonable

to assume that all agents simply see the fastest route, and all choose it, causing

heavy tra\_c and slowing the route down. Autonomous vehicles will have access to much

better information than humans, so maintaining heterogeneous decision making will be

vital to evenly distributing tra\_c to minimise travel times.

Lastly it is cleat from the work of Chen et al. (2017) that links behave di\_erently with

mixed tra\_c than with human tra\_c. Autonomous vehicles lead to high capacities which

should reduce travel times. However the highlighted work did not explore this e\_ect with

ABMS.

18

Project Report William Kane

An few areas of research emerge from this literature review. Most obviously is the need for

AMBS of day to day tra\_c demand and ow dynamics with mixed human and autonomous

vehicles on which there has been little research compared to the human-only case. However

the rationality of autonomous vehicles should be explored, especially the idea of making

them irrational like a human. The rationality of the human tra\_c and the e\_ect this has

on the overall mixed tra\_c is also work exploring.

The goal of this report is therefore to explore with ABMS day to day tra\_c demand and

ow dynamics with mixed human and autonomous vehicles, and the e\_ects of information

and rationality.

19

Project Report William Kane

4 Methodology/ The Model

4.1 Overview

It is assumed that a number of vehicles N wish to travel on a road network. These vehicles

may use human driven or autonomous, the number of each is given by Nhv and Nav for

HVs and AVs respectively, and Nhv + Nav = N. Each vehicle is an independent agent.

Each agent has separate knowledge of travel times, an origin and destination between

which they want to travel, and choose their route independently of other agents. This

constitutes each agent's belief, desire, and intention.

The simulation allows all N agents to make their journey on a given network. On one

day each agent travels from their origin to their destination. Within-day dynamics is not

considered, travel times assume all agents that travel on a road do so simultaneously.

This assumption makes modelling easier, and is suitable for simulating events such as

rush-hour when many drivers take to the roads on their daily commute.

Each agent has a degree of rationality, represented as preference towards their perceived

shortest route. Each driver is also able to look up the previous day's travel times on any

route using the ATIS, and can trust it as much as they want. Error terms are also used on

perceived travel times for a route they have travel. It is assumed AVs are more rational

than HVs, have better memory, and update their memory on all routes available at the

end of the day, while HVs only update travel times they actually experience, thus only

the route they travel on.

At the start of the day, each driver chooses a route randomly. There is a \_xed probability

\_ of them deciding to try a new route today, if so then the probability of each alternative

route is a function of their perceived travel times on each of the routes, such that faster

routes are preferred. Perceived travel times balance both memory and ATIS information.

The rationality parameter \_ controls the distribution, higher thetas make the shortest

route more likely. Once all agents have picked a route, the numbers are counted per link

so that travel times can be calculated. Busy links lead to slower travel times.

At the end of the day each agent adjusts its memory. HVs update their list of remembered

travel times for each link they travelled on, with error. AVs update for all links in the

network as they have access to better information, there is error on this too but by default

it is 0. It is now possible to run the simulation again for another day.

The parameters in the model, with their default values, are shown in table 3. These values

can be varied to study their e\_ects.

When choosing a new route, the probability for agent i choosing route j, with given \_,

and \_ is given by equation 12. Note for an AV, \_ = 1. The probability of not changing

route, and using the same as yesterday is 1􀀀\_ for HVs. This is the Gumbel distribution

20

Project Report William Kane

Table 3: Model parameters and default values

Parameter Default Value Description

D 500 The number of days the simulation is ran for

Nhv;Nav 500; 500 The number of HVs and AVs

respectively

ehv; eav 5; 0 The standard deviation error in perceived travel times for

HVs and AVs respectively. Errors are normal

random variables with mean 0

Lhv; Lav 3; 1000 the memory length in days for HVs and AVs respectively

\_hv; \_av 0:5; 1 The rationality of HVs and AVs respectively

\_hv; \_av 0; 0 The ATIS bias of HVs and AVs respectively

\_ 0.5 The probability of a HV choosing a new route on a given

day. There is not such value for AVs which always

consider all routes

often used in literature such as Zheng et al. (2013) and Liu & Huang (2007).

pi;j(d) = \_

Pexp (􀀀\_ti;j)

k exp (􀀀\_ti;k)

(12)

ti;j is the perceived travel time of that route, given by the average of the the agents memory

mj , mixed with ATIS information giving the exact travel times from the previous day cj ,

and is given in equation 13. This equation is based of Liu & Huang (2007).

ti;j = \_mj + (1 􀀀 \_)cj (13)

The travel time cl on each link l is a function of the links tra\_c count vl, capacity Yl,

and free-ow travel time c0l

(the travel time with no tra\_c) given in equation 2, the BPR

function used through literature.

cl = c0l

(1 + 0:15(vl=Yl)4) (2)

However as the capacity of a link increases with the proportion of AVs to AVs + HVs, Yl

is a function of Nhv and Nav given in equation

Yl = f(\_; ") =

Yl;0

1 􀀀 \_"

(14)

21

Project Report William Kane

where

\_ =

Nav

Nhv + Nav

(15)

and

" =

(

1 􀀀 􀀀

\_

\_A􀀀

n + \_R􀀀1

n

\_

; if 0 \_ \_ < 1

1 􀀀 􀀀 \_A􀀀

n ; if \_ = 1

(16)

The constraints on ; \_A and \_R are \_R \_ 1; < 1; \_A \_ , " < 1, and \_R; ; \_A \_ 0.

Under these constraints, the values chosen are \_A = 0:9; \_R = 1:2; = 0:75. The platoon

length n = 5 is used.

4.2 Program

The simulation is written in python, with classes for HVs, AVs, links, and networks.

The simulation ran in a Jupyter Lab interactive environment, with class and function

de\_nitions in imported modules. The program is structured as follows.

1. Global initialisation: set values for the model parameters.

2. De\_ne a list of links, with a start, end, capacity, and free ow travel time.

3. De\_ne a network from this list of links.

4. De\_ne a list of agents, HVs and AVs each with an origin and destination.

5. Simulate day 1. Each agent has a method to run on day 1. Their OD is sent to

the network to get all available routes, their memory is initialised as the free ow

travel time, plus a large random error, given by 10 times their normal error standard

deviation. They then pick a route.

6. The network updates. Counts per links are made and travel times are calculated.

7. Data is saved on the counts per link and per route, using the state of the network,

and by looping through all agents and recording their route.

8. The drivers amend their memory. HVs remember up to Lhv travel times per link

and add their experienced travel time plus error for each link from the previous trip

to their memory. AVs do the same but for all links. They then pick a new route.

9. Loop through 6,7,8 for D Days.

The program is coded in an object orientated manor, with the network object built from

link objects. This makes the model versatile and it can be quickly applied to any network.

Link objects contain a method to calculate travel times, with capacity adjustment

depending on the number of AVs using the link. The network object contains methods to

22

Project Report William Kane

\_nd all paths between an origin and destination, and a method to count how many agents

used each link then calculate travel times with each links travel time method.

The main limitations of this program are: the path \_nding algorithm used only works on

\_nite acyclic digraphs (no loops or in\_nite paths allowed); and if links are signi\_cantly

over capacity, thus travel times are high enough, the denominator of equation 12 used

when agents pick a route may come out as zero due to \_nite resolution, crashing the

simulation. Two prevent the second limitation from happening, the total capacity of the

network between any OD is not lower than the number of agents.

4.3 Usage

The model was used to run simulations with varied initialisation parameters to study their

e\_ects. The simulations ran are listed below. All other variables are held at their default

values.

• (Nhv;Nav) = (1000; 0); (900; 100); :::(100; 900); (0; 1000)

• \_hv = 0:01; 0:05; 0:1; :2; :3; :4; :5; 1; 2; 3

• \_av = 0:01; 0:05; 0:1; :2; :3; :4; :5; 1; 2; 3

• Lhv = 1; 2; 3; 4; 10; 30; 50; 100; 250

• Lav = 1; 2; 3; 4; 10; 30; 50; 100; 250

• \_hv; \_hv = (0; 0); (0; :5); (0:5; 0); (:5; :5)

For each simulation, route count data per day was saved to a \_le. These datasets were

analysed in a separate notebook.

The \_rst network used for analysis is Test Network 1, shown in \_gure 7, with capacities

and free ow travel times in table 4. All agents on this network wish to travel from node

1 to node 9, there are 6 possible paths for this, labelled route 0 to route 5 and detailed

in table 5. This network is taken from Wei et al.'s paper Wei et al. (2014), with the only

di\_erence being each capacity is doubled.

23

Project Report William Kane

Figure 7: Nodes and links in Test Network 1

Table 4: Capacities and free ow travel times in Test Network 1

Link C0

l Yl Link C0

l Yl

(1-2) 20 720 (5-6) 12 360

(2-3) 12 720 (4-7) 15 480

(1-4) 15 480 (58) 10 300

(2-5) 12 360 (6-9) 30 720

(3-6) 12 720 (7-8) 15 480

(4-5) 10 300 (8-9) 15 480

Table 5: Paths between nodes 0 and 9 in Test Network 1.

Path Links

Route 0 (1-2), (2-3), (3-6), (6-9)

Route 1 (1-2), (2-5), (5-6), (6-9)

Route 2 (1-2), (2-5), (5-8), (8-9)

Route 3 (1-4), (4-5), (5-6), (6-9)

Route 4 (1-4), (4-5), (5-8), (8-9)

Route 5 (1-4), (4-7), (7-8), (8-9)

24

Project Report William Kane

5 Results

5.1 Default Simulation

The simulation was ran on Test Network 1 with all the default parameters to access its

performance and behaviour. All agents travelled for node 1 to node 9 for 500 days, the

number of agents choosing each route is recorded each day and plotted in \_gure 8. By

taking the mean of the last 250 days, it is possible to estimate the converge point for each

route, these means are shown in table 6 and marked with black dashed lines in \_gure 8.

In \_gure 9 the travel time for each route are shown over the course of the simulation and

the average travel times for the last 250 days are also show in table 6. This simulation

was ran multiple times and these results can be considered typical.

Table 6: Apparent point of converge for each route on Test Network 1

Mean: last 250 days

Path Tra\_c Flow Travel Time

Route 0 305.2 85.4

Route 1 112.1 87.7

Route 2 123.4 86.3

Route 3 74.9 87.2

Route 4 172.3 85.8

Route 5 212.1 86.0

5.2 E\_ects of AV Proportion

To study the e\_ects of the proportion of AVs to HVs, eleven simulations where ran varying

the amount of each, with (Nhv;Nav) = (1000; 0); (900; 100); :::(100; 900); (0; 1000). To

visualise the data, two \_gures are drawn. Figure 10 shows a sample of four simulations

with tra\_c on routes 0 and 2 displayed. Figure 11 shows the standard deviation of tra\_c

ow per route over the previous \_ve days, for each simulation ran. This gives measure

of stability around the convergence values. Each route converged to approximately the

same values as the default run.

5.3 E\_ects of \_hv

The simulation was ran 10 times with default parameters but varying \_hv through

0:01; 0:05; 0:1; 0:2; 0:3; 0:4; 0:5; 1; 2; 3

In \_gure 12, four simulations are shown for references, with the tra\_c on routes 0 and

2 shown over the course of the simulation. In \_gure 13 the 5 day standard deviation is

shown for each \_hv value.

25

Project Report William Kane

Figure 8: Simulation results from default parameters

Figure 9: Route Travel times with default parameters

5.4 E\_ects of \_av

The simulation was ran 10 times with default parameters but varying \_av through

0:01; 0:05; 0:1; 0:2; 0:3; 0:4; 0:5; 1; 2; 3

26

Project Report William Kane

Figure 10: Results of simulations with di\_erent Nav;Nhv

Figure 11: Results of simulations with di\_erent Nav;Nhv

In \_gure 14, four simulations are shown for references, with the tra\_c on routes 0 and 2

shown over the course of the simulation. In \_gure 15 the 5 day standard deviation is shown

27

Project Report William Kane

Figure 12: Results of simulations with di\_erent \_hv

Figure 13: Results of simulations with di\_erent \_hv

for each \_av value. Figure 16, the tra\_c on route 0 is shown in detail for \_av = 0:01; 3.

28

Project Report William Kane

Figure 14: Results of simulations with di\_erent \_av

Figure 15: Results of simulations with di\_erent \_av

5.5 E\_ects of Lhv

To study the e\_ects of Lhv the simulation was ran 10 time with the values of

Lhv = 1; 2; 3; 4; 10; 30; 50; 100; 250

29

Project Report William Kane

Figure 16: Results of simulations with di\_erent \_av

with all other parameters held at their defaults.

In \_gure 17, four simulations are shown for references, with the tra\_c on routes 0 and

2 shown over the course of the simulation. In \_gure 18 the 5 day standard deviation is

shown for each Lhv value.

5.6 E\_ects of Lav

To study the e\_ects of Lav the simulation was ran 10 time with the values of

Lav = 1; 2; 3; 4; 10; 30; 50; 100; 250

with all other parameters held at their defaults.

In \_gure 19, four simulations are shown for references, with the tra\_c on routes 0 and

2 shown over the course of the simulation. In \_gure 20 the 5 day standard deviation is

shown for each Lav value.

5.7 E\_ects of \_

To study the e\_ects of \_ the simulation was ran 10 time with the values of

\_hv; \_hv = (0; 0); (0; :5); (0:5; 0); (:5; :5)

30

Project Report William Kane

Figure 17: Results of simulations with di\_erent Lhv

Figure 18: Results of simulations with di\_erent Lhv

with all other parameters held at their defaults.

In \_gure 21, all four simulations are shown for references, with the tra\_c on routes 0 and

31

Project Report William Kane

Figure 19: Results of simulations with di\_erent Lav

Figure 20: Results of simulations with di\_erent Lav

2 shown over the course of the simulation.

32

Project Report William Kane

Figure 21: Results of simulations with di\_erent \_

6 Discussion

6.1 Overall Behaviour

The model presented in this report shows consistency with models on Test Network 2 in

literature.

As can be seen in \_gure 8 the ow on each of the six routes converged quickly, in under

100 days after which all ows remained stable around the mean of the \_nal 250 days.

They only variations visible around the mean appears to be random. This randomness is

baked into the model as when a HV decided to try another route, with a probability of

a half, it cannot choose the route it was on before and must explore a new route. All six

routes reached equilibrium with travel times of approximately 86 seconds.

The model is fast to run, and can be completed in 30 seconds on an Intel i7-9750H. The

performance of this simulation was useful to run loops over the simulation with varying

parameters, allowing for fast data collection to study the e\_ects of the model parameters.

6.2 AV Proportion

As can be seen in \_gure 11, an increase in the proportion of AVs in the tra\_c leads to

less variation and a more stable equilibrium in the long run, but not at \_rst.

33

Project Report William Kane

When the tra\_c only consists of HVs an oscillatory pattern begins to emerge as can be

see in \_gure 10. The standard deviation fall quickly as each HVs build up a memory of

the tra\_c on each route but as the simulation continues the emerging oscillatory pattern

gets ampli\_ed, stabling at a higher standard deviation than when the ow \_rst converges

to the long-term means. Both the smaller standard deviation and the applied pattern

center around the same means, so even as the stability lessens, the convergence values

remain the same and similar to the default simulation.

One possible explanation for this is, as the memory length of each HV here is 3, an odd

number, each agent will have memory in the form (high; low; high) or (low; high; low),

with averages coming out as fairly high or fairly low. This causes the expected travel

times on each routes to oscillate with could explain this behaviors. This is explored in

the discussion on human memory length later..

The hypnosis that memory length is causing this behaviour \_ts with the AV only case

where the AV memory length is greater than the length of the simulation, at 1000 days.

AVs see expected travel times which are the average each route has experienced so far, for

all routes including once which they haven't used. This introduces stability, particularly

over time as each addiction to memory will have less of an impact on the mean of the

memory as the list grows.

AVs however experience a lot of instability in the \_rst 30 days. This could be due to

a higher rationality parameter \_av combined with a homogeneous memory as all agents

update their memory for all routes. This would be expected to cause heavy ow on the

fasted route, with all drivers agree is the most likely choice. This is explored more in the

discussion on rationality.

6.3 Rationality

Both the e\_ects of \_hv and \_av make up the data collected to explore rationality. \_ is the

rationality parameter, a higher value leads to a higher probability of choosing a faster

perceived route, by equation 12.

In \_gure 12 and \_gure 13, convergence happens quickly over the \_rst 20 days, then continues

slower over the next 20, reaching the emergent state by day approximately 40.

The standard deviation increases with \_hv but not by a huge amount. Even when the

rationality of the HVs is above that of the AVs they are travelling with, the convergent

states are reached with only mind oscillation.

Figure 14 and \_gure 15 show the same experiment performed on AVs. In this case the

e\_ect is much greater. As \_av increases, oscillatory behaviour develops and grows to

much greater aptitudes than with the HV case. Flow over all routes when \_av = 0:01

is particularly stable, although it does appear to be at a lower level. By calculating the

34

Project Report William Kane

average of the \_nal 250 days on route 0 ,the convergence point is found be 228:0, below

the default case of 305.2. In the cases of low \_av convergence also happens signi\_cantly

faster than the default case.

When \_av = 3 the tra\_c on route 0 takes nearly 200 days to converge, and shows high

amplitude random oscillations around a higher mean of 341.4. Due to the homogeneous

memory of AVs, it is expected that higher rationality would cause too much tra\_c on

the fasted routes, and route 0 is the fastest in free ow. This would explain the tra\_c on

route 0 but not the oscillatory behaviour.

It is clear that overly rational AVs may lead to worse tra\_c, without more intelligent

decision making algorithms that understand these e\_ects and attempt to prevent them.

6.4 Memory

Figure 17 shows the results when HV memory is 1, 4, 50, and 250, while \_gure 18 shows

the e\_ects memory length has on the standard deviation.

The standard deviations suggest HV memory length doesn't have a huge impact on convergence

speed, with lower shorter values of Lhv converging slightly faster higher ones.

Higher Lhv also have a higher standard deviation once a steady state has been reached.

The hypnosis that the odd values of 3 for Lhv was causing oscillatory behaviour seem not

to hold, as oscillatory behaviour continues to be seen as is highlighted in \_gure 17 for

Lhv = 4.

The e\_ects of memory for AVS is drastic. In \_gure 19 it can be seen that small memory

lengths lead to high amplitude oscillatory behaviour. The Lav case appears to show a

approximately stable 5-cycle. This could be due to agents forgetting the travel times

from 5 days ago causing them to repeat them selves. Agents with homogeneous memory

appear to need complete memory in order to stabilise without oscillations.

To test this an extra simulation was ran with Lav = 4 and Nav = 1000 with no HVs. A

portion on the results showing the tra\_c on route 0 on between day 20 and day 40 is

shown in 22.

Memory can also include information from the the Advanced Traveller Information System.

The ATIS mixes the previous days information for all routes with the agents memory.

This could have the e\_ect of blending the sort memory length case with the longer case.

From studying \_gure 21 it can be seen the the ATIS has small di\_erence to HVs but

causes high amplitude oscillations in AVS. This agrees with the hypnosis of blending

cases. The small e\_ect on HVs is likely due to their short memory, so blending Lhv = 4

and 1 makes no di\_erence, however the ATIS causes HVS con convergence much slower,

taking approximately 40 days instead of 20.

35

Project Report William Kane

Figure 22: Section of results from extra simulation on Lav = 4, results between days 20,

and 40

6.5 Comparison with literature

Comparing with literature, in Wei et al. (2014) it was found that longer memory led to

more stability, however in their model memory was a weighted average of past experiences.

The HV case in this report's model disagrees with this, with higher Lhv leading to more

oscillations. The AV case agrees with the literature as a better memory leads to better

convergence and less oscillation.

Although this report's model has no learning rate, l from Wei et al. (2014) can be thought

of as a rationality parameter. Both the AV and HV cases agree with their results as higher

rationality leads to less stable steady states and more oscillation.

The simple ATIS in this report's model shown that providing HVs with accurate information

does make ow convergence occur slower, but it still converges, unlike the oscillatory

pattern found by Liu & Huang (2007). AVs already have access to accurate information

as they have no error. However the ATIS prevents AVS from using their average travel

times and instead favours the previous days tims, this causes powerfully oscillations and

the instability found in Liu & Huang (2007).

36

Project Report William Kane

7 Conclusions

From the Simulations and analysis in this report, it can be seen that autonomous vehicles

could improve tra\_c ow but only when implemented correctly.

An agent based model was built with separate human and autonomous vehicles. HVs

chose a route based on their memory from experiences while AVS learned from collective

experience. Humans had error in their perception while AVs did not. Agents chose a

route with a higher probability for faster routes, with AVs preferring faster routes more

than HVs.

The agents drove on a grid network from a common origin to a common destination.

They had choice of routes which shared links with each other. They began with vague

knowledge of travel times, with a lot of error, and learned from experience. They drove

500 times, learning every time.

Introducing AVs does lead to initial instability, which get worse as AV penetration gets

higher, but this is only initial as the AVs learn the network. Higher AV penetration does

improve travel times towards the end of the simulation. In reality when AVs are deployed,

they will likely already have access to information, and a gradual introduction shouldn't

lead to the instability seen as the simulation begins.

It is also clear that for AVs to mix well with humans, arti\_cial irrationally could be

extremely useful. By limiting the information available to the AVs while choosing a

route, and preventing them all from choosing the fastest route, the tra\_c will spread out

more evenly.

Preventing overcrowding on fast routes will be one of the important tasks to master when

developing autonomous vehicles, but performed successfully could lead to improved travel

times for all road users.

37

Project Report William Kane

Bibliography

Ackerman, E. (2012), `Study, intelligent cars could boost highway capacity by 273 percent',

IEEE Spectrum .

Bertoncello, M. & Wee, D. (2015), `Ten ways autonomous driving could rede\_ne the

automotive world', McKinsey And Company 6.

Bonabeau, E. (2002), `Agent-based modeling: Methods and techniques for simulating

human systems', Proceedings of the National Academy of Sciences 99, 7280.

Cascetta, E. (1989), `A stochastic process approach to the analysis of temporal dynamics

in transportation networks', Transportation Research Part B: Methodological 23, 1{17.

Chen, D., Ahn, S., Chitturi, M. & Noyce, D. A. (2017), `Towards vehicle automation:

Roadway capacity formulation for tra\_c mixed with regular and automated vehicles',

Transportation Research Part B: Methodological 100, 196{221.

Cross, J. G. (1973), `A stochastic learning model of economic behavior', The Quarterly

Journal of Economics 87, 239.

Daganzo, C. F. & She\_, Y. (1977), `On stochastic models of tra\_c assignment', Trans-

portation Science 11, 253{274.

Ford, L. R. & Fulkerson, D. R. (1956), `Maximal ow through a network', Canadian

Journal of Mathematics 8, 399{404.

Frank, M., Wolfe, P. et al. (1956), `An algorithm for quadratic programming', Naval

research logistics quarterly 3, 95{110.

Gardner, M. (1970), `Mathematical games', Scienti\_c American 223, 120{123.

Hazelton, M. L. & Parry, K. (2016), `Statistical methods for comparison of day-to-day

tra\_c models', Special issue Day-to-Day Dynamics in Transportation Networks 92, 22{

34.

Horowitz, J. L. (1984), `The stability of stochastic equilibrium in a two-link transportation

network', Transportation Research Part B: Methodological 18, 13{28.

Jungnickel, D. (2013), Graphs, Networks and Algorithms, 4 edn, Springer Berlin Heidelberg.

Lee, S., Son, Y.-J. & Jin, J. (2010), `An integrated human decision making model for

evacuation scenarios under a bdi framework', ACM Trans. Model. Comput. Simul. 20.

38

Project Report William Kane

Li, R., Liu, X. & Nie, Y. M. (2018), `Managing partially automated network tra\_c ow:

E\_ciency vs. stability', Transportation Research Part B: Methodological 114, 300{324.

Liu, T. & Huang, H. (2007), Multi-agent simulation on day-to-day route choice behavior,

Vol. 5, pp. 492{498.

Macal, C. M. & North, M. J. (2005), Tutorial on agent-based modeling and simulation,

p. 14 pp.

Manual, T. A. (1964), `Urban planning division', US Department of Commerce, Wash-

ington DC .

Nakayama, S., Kitamura, R. & Fujii, S. (2001), `Drivers' route choice rules and network

behavior: Do drivers become rational and homogeneous through learning?', Transporta-

tion Research Record 1752, 62{68.

Ntousakis, I. A., Nikolos, I. K. & Papageorgiou, M. (2015), `On microscopic modelling of

adaptive cruise control systems', Transportation Research Procedia 6, 111{127.

Peeta, S. & Yang, T.-H. (2003), `Stability issues for dynamic tra\_c assignment', Auto-

matica 39, 21{34.

Prashker, J. N. & Bekhor, S. (2004), `Route choice models used in the stochastic user

equilibrium problem: A review', Transport Reviews 24, 437{463.

Wardrop, J. G. (1952), `Road paper. some theoretical aspects of road tra\_c research.',

Proceedings of the institution of civil engineers 1, 325{362.

Watling, D. & Hazelton, M. L. (2003), `The dynamics and equilibria of day-to-day assignment

models', Networks and Spatial Economics 3, 349{370.

Wei, F., Ma, S. & Ning, J. (2014), `A day-to-day route choice model based on reinforcement

learning', Mathematical Problems in Engineering 2014.

Zhang, L. (2011), `Behavioral foundation of route choice and tra\_c assignment', Trans-

portation Research Record: Journal of the Transportation Research Board 2254, 1{10.

Zheng, H., Son, Y.-J., Chiu, Y.-C., Head, L., Feng, Y., Xi, H., Kim, S. & Hickman, M.

(2013), `A primer for agent-based simulation and modeling in transportation applications'.

39

Project Report William Kane

Appendix 1: Python Code

Simulation Script

1 from agents import \*

2 from environment import \*

3 import pickle

4

5 # Init variables

6 hv = 500 # n of HVs

7 av = 500 # n of AVs

8 N = 500 # n of Days

9 orig = '1'

10 dest = '9'

11 hv\_err = 5 # error term on HV time perception ~N(0, hv\_err)

12 hv\_theta = .5 # rationality

13 hv\_beta = .5 # prob of change route

14 hv\_len = 3 # Memory lenth

15 hv\_atis\_bais = 0 # bias\*prevTT + (1-bais)\*memTT

16 av\_err = 0

17 av\_theta = 1

18 av\_len = 1000

19 av\_atis\_bias = 0

20

21 # Data save paths

22 route\_dir = f"data/sim-ROUTES-N{N}-hv{hv}at{hv\_err}\_{hv\_theta}\_{hv\_beta}\_{hv\_len}\_{

hv\_atis\_bais}-av{av}at{av\_err}\_{av\_theta}\_{av\_len}\_{av\_atis\_bias}.pickle"

23 roads\_dir = f"data/sim-ROADS-N{N}-hv{hv}at{hv\_err}\_{hv\_theta}\_{hv\_beta}\_{hv\_len}\_{

hv\_atis\_bais}-av{av}at{av\_err}\_{av\_theta}\_{av\_len}\_{av\_atis\_bias}.pickle"

24

25 # Square Network

26 roads = [Road('1', '2', 720, 20), Road('2', '3', 720, 12), Road('1', '4', 480, 15),

Road('2', '5', 360, 12),

27 Road('3', '6', 720, 12), Road('4', '5', 300, 10), Road('5', '6', 360, 12), Road('4',

'7', 480, 15),

28 Road('5', '8', 300, 10), Road('6', '9', 720, 30), Road('7', '8', 480, 15) ,Road('8',

'9', 480, 15)]

29

30 # Make Network from roads

31 network = Network(roads)

32

33 # Make drivers

34 drivers = [HV(orig, dest, err = hv\_err, theta = hv\_theta, beta = hv\_beta, L = hv\_len)

for i in range(0, hv)]

35 if av > 0:

36 drivers = drivers + [AV(orig, dest, theta = av\_theta, err = av\_err, L = av\_len,

atis\_bias = av\_atis\_bias) for i in range(0, av)]

37

38 # Day 1

40

Project Report William Kane

39 for driver in drivers:

40 driver.learn(network)

41 network.update(drivers)

42

43 # Make logs

44 count\_log = pd.DataFrame(columns = [f'Road{i}' for i in range(len(network.roadlist))])

45 route\_log = pd.DataFrame(columns = [f'Route{i}' for i in range(len(drivers[0].routes))

])

46

47 # Save day 1 data

48 count\_log.loc[0] = [road.count for road in network.roadlist]

49 route\_count = [0 for route in range(len(drivers[0].routes))]

50 for driver in drivers:

51 # Add one to route i, if driver took root i

52 route\_count[driver.i] = route\_count[driver.i] + 1

53 route\_log.loc[0] = route\_count

54

55 # Day > 1 Loop

56 for i in range(1,N):

57

58 # Simulate

59 for driver in drivers:

60 driver.drive(network)

61 network.update(drivers)

62

63 # Save data

64 count\_log.loc[i] = [road.count for road in network.roadlist]

65 route\_count = [0 for route in route\_log.keys()]

66 for driver in drivers:

67 route\_count[driver.i] = route\_count[driver.i] + 1

68 route\_log.loc[i] = route\_count

69

70 # Export data to file for analysis

71 pickle.dump(route\_log, open(route\_dir, "wb" ))

72 pickle.dump(count\_log, open(roads\_dir, "wb" ))

41

Project Report William Kane

Agent Module

1 # Agent Definitions

2 # Generic driver class, then HV and AV classes

3

4 import pandas as pd

5 from random import uniform, gauss

6 from math import exp

7

8

9 # Probablity of route from perceived travel time list

10 def problist(plist, beta, theta):

11 plist = [p for p in plist]

12 top = [exp(-(theta) \* p) for p in plist]

13 sums = sum(top)

14 if sums == 0:

15 raise Exception("Probablites came out as 0 as e^-theta\*p is too small. The

Roads are probably over capacity.")

16 qlist = [beta \* top[i] / sums for i in range(0, len(top))]

17 return qlist

18

19

20 # Generic Vehicle Class

21 class Driver:

22 type = "GV"

23

24 def \_\_init\_\_(self, origin, destination, beta = .5, theta = .5, L = 3, err = .1,

atis\_bias = 0.5):

25 self.origin = origin

26 self.destination = destination

27 self.beta = beta

28 self.theta = theta

29 self.L = L

30 self. err = err

31 self.bias = atis\_bias

32

33 def \_\_str\_\_(self):

34 return f"{self.type}:{(self.origin, self.destination)}"

35

36 # Driver learns network

37 def learn(self, network):

38 # init ett = freeflow tt + error

39 self.memory = {road:[float(road.freeflow) + gauss(0, self.err\*10)] for road in

network.roadlist}

40

41 # Get routes and ETTs

42 self.routes = network.routes(self.origin, self.destination)

43 ett = []

44 for route in self.routes:

42

Project Report William Kane

45 tt = 0

46 for road in route:

47 tt = tt + (1-self.bias) \* self.memory[road][0] + self.bias \* road.

tt(network)

48 ett.append(tt)

49

50 # Init route choose

51 probs = problist(ett, 1, self.theta) # beta = 1 as must pick new route

52 rand = uniform(0, 1)

53 i = 0

54 while rand > sum([probs[j] for j in range(i + 1)]):

55 i += 1

56 self.route = self.routes[i]

57 self.i = i

58

59 def drive(self, network):

60 # Update ETTs in roads

61 for road in self.route:

62 self.memory[road].append(road.tt(network) + gauss(0, self.err))

63 if len(self.memory[road]) == self.L + 1:

64 self.memory[road].pop(0) # Only up to L days memory

65

66 # Update ETTs in routes

67 ett = []

68 for route in self.routes:

69 tt = 0

70 for road in route:

71 tt = tt + (1-self.bias)\*(sum(self.memory[road]) / len(self.memory[road

])) + self.bias\*road.tt(network)

72 ett.append(tt)

73

74 # Choose next route

75 ett.pop(self.i)

76 probs = problist(ett, self.beta, self.theta) # p of routes iff change

77 p\_same = 1 - self.beta # p no change

78 probs.insert(self.i, p\_same)

79 rand = uniform(0, 1)

80 i = 0

81 while rand > sum([probs[j] for j in range(i + 1)]):

82 i += 1

83 self.route = self.routes[i]

84 self.i = i

85

86 def display(self):

87 df = pd.DataFrame(self.memory.keys(), columns=["Road"])

88 df['Memory'] = list(self.memory.values())

89 etts = []

90 for road in self.memory.keys():

91 ett = sum(self.memory[road]) / len(self.memory[road])

43

Project Report William Kane

92 etts.append(ett)

93 df['ETT'] = etts

94 print(f"Last route: route {self.i}:{self.route}")

95 return df

96

97

98 # Autonomous class

99 class AV(Driver):

100 type = 'AV'

101

102 def \_\_init\_\_(self, origin, destination, theta = 1, L = 1000, err = 0, atis\_bias =

.5):

103 super().\_\_init\_\_(origin, destination, theta = theta , L = L, err = err,

atis\_bias = atis\_bias)

104

105

106 # Edited to update ALL roads

107 def drive(self, network):

108

109 for road in network.roadlist:

110 self.memory[road].append(road.tt(network))

111 if len(self.memory[road]) == self.L + 1:

112 self.memory[road].pop(0)

113

114 ett = []

115 for route in self.routes:

116 tt = 0

117 for road in route:

118 tt = tt + (1-self.bias)\*(sum(self.memory[road]) / len(self.memory[road

])) + self.bias\*road.tt(network)

119 ett.append(tt)

120

121 # Choose next route from ALL roads

122 probs = problist(ett, 1, self.theta)

123 rand = uniform(0, 1)

124 i = 0

125 while rand > sum([probs[j] for j in range(i + 1)]):

126 i += 1

127 self.route = self.routes[i]

128 self.i = i

129

130

131 class HV(Driver):

132 type = 'HV'

44

Project Report William Kane

Environment Module

1 # Envronment Definitions

2 # From roads to Networks, plus some functions

3 # Function to calc tt from count, ff, capacity

4 # Function to calc av gain from params

5

6 import pandas as pd

7

8 # Travel time fn

9 def traveltime(count, freeflow, capacity):

10 tt = freeflow \* (1 + 1.15 \* ((count / capacity) \*\* 4))

11 return tt

12

13

14 # AV gain fn

15 def avgain(g, ba, br, n, a):

16 """ g: av spacing

17 ba: av spacing bebind hv

18 br: hv spacing behind av

19 n: platoon lenth

20 all as propotions of hv-hv spacing

21 """

22 if a == 0:

23 e = 1 - g - (ba - g) / n

24 else:

25 e = 1 - g - ((ba - g) / n + (br - 1) / n)

26 return e

27

28

29 # Road: From a to b with capacity c, free flow f, and count, av count, and traveltime

()

30 class Road:

31 def \_\_init\_\_(self, a, b, c, f):

32 self.start = a

33 self.end = b

34 self.capacity = c

35 self.freeflow = f

36 self.count = 0

37 self.av\_count = 0

38

39 def tt(self, network):

40 # Adjust capacity for AVs, based on network's AV params

41 a = self.av\_count / self.count if self.count != 0 else 0

42 e = avgain(network.g, network.br, network.ba, network.n, a)

43 capacity = self.capacity / (1 - a \* e)

44 tt = traveltime(self.count, self.freeflow, capacity)

45 return tt

46

45

Project Report William Kane

47 def \_\_str\_\_(self):

48 return f'Road({self.start}->{self.end})'

49

50 \_\_repr\_\_ = \_\_str\_\_

51

52

53 # Network: Built from a list of roads

54 class Network:

55 """ Indexes roads by order in init list

56 """

57

58 def \_\_init\_\_(self, roads, ba=0.9, br=1.2, g=0.75, n=5):

59 self.roadlist = roads

60 self.nroads = len(self.roadlist)

61 self.ba = ba

62 self.br = br

63 self.g = g

64 self.n = n

65

66 def display(self):

67 # Pandas DataFrames are human readable so

68 db = pd.DataFrame(self.roadlist, columns=["Road"])

69 db['Origin'] = [road.start for road in self.roadlist]

70 db['Destination'] = [road.end for road in self.roadlist]

71 db['Capacity'] = [road.capacity for road in self.roadlist]

72 db['Free Flow'] = [road.freeflow for road in self.roadlist]

73 db['Count'] = [road.count for road in self.roadlist]

74 db['TT'] = [road.tt(self) for road in self.roadlist]

75 return db

76

77 # path finding alg

78 def routes(self, origin, destination):

79 routes = []

80 # Starting roads

81 explore = [road for road in self.roadlist if road.start == origin]

82 # check if start roads reach destination

83 closed = [[road] for road in explore if road.end == destination]

84 routes = routes + closed

85 # Open for exploration, looped until done

86 opn = [road for road in explore if road not in closed]

87 explore = []

88 for entry in opn:

89 next = [[entry, road] for road in self.roadlist if road.start == entry.end

]

90 explore = explore + next

91 while len(explore) > 0:

92 for path in explore:

93 explore = [entry for entry in explore if entry is not path]

94 end\_road = path[-1]

46

Project Report William Kane

95 end\_explore = [road for road in self.roadlist if road.start ==

end\_road.end]

96 closed = [path + [road] for road in end\_explore if road.end ==

destination]

97 routes = routes + closed

98 opn = [path + [road] for road in end\_explore if road not in closed]

99 explore = explore + opn

100 return routes

101

102 # Road to index type change

103 def index(self, road):

104 index = self.roadlist.index(road)

105 return index

106

107 def update(self, drivers):

108 for road in self.roadlist:

109 count = 0

110 av\_count = 0

111 for driver in drivers:

112 if road in driver.route:

113 count += 1

114 if driver.type == 'AV':

115 av\_count += 1

116 road.count = count

117 road.av\_count = av\_count

118

119

120 if \_\_name\_\_ == '\_\_main\_\_':

121 roads = [Road('1', '12', 1000, .02), Road('1', '5', 1000, .02), Road('4', '5',

1000, .02),

122 Road('4', '9', 1000, .02), Road('5', '9', 1000, .02), Road('5', '6',

1000, .02),

123 Road('9', '10', 1000, .02), Road('9', '13', 1000, .02),

124 Road('6', '7', 1000, .02), Road('12', '8', 1000, .02), Road('10', '11',

1000, .02),

125 Road('13', '3', 1000, .02), Road('12', '6', 1000, .02), Road('6', '10',

1000, .02),

126 Road('7', '11', 1000, .02), Road('7', '18', 1000, .02), Road('11', '2',

1000, .02),

127 Road('11', '3', 1000, .02), Road('8', '2', 1000, .02)]

128 network = Network(roads)

129 print(network.routes('1', '2'))

47

Project Report William Kane

Appendix 2: Gantt Chart

48