NTNU – Trondheim
Norwegian University of
Science and Technology

Numerical Solutions of Traffic Flow on Networks

Using the LWR-Model and the Godunov Scheme

Bjørnar Dolva Bergersen

Master of Science in Mathematics
Submission date: May 2014
Supervisor: Helge Holden, MATH

Norwegian University of Science and Technology
Department of Mathematical Sciences

Abstract/Sammendrag

This paper shows how to create a simulationtool for traffic flow in a network using the Lighthill–Witham–Richards model and the Godunov scheme. First some basic rules about conservation laws are described and how to solve them using the method characteristics. This leads to the notion of weak solutions which can be solved by shock- and rarefractions-solutions. This is then used to describe how traffic behaves on a single road by using the LWR-model. The behavior of traffic at junctions is discussed, more specifically how to find the maximum flux through a junction when we deal with different amount of incoming and outgoing roads. The paper gives different examples of numerical solution methods to conservation laws, which gives motivation for the Godunov scheme. A numerical scheme using the LWR-model and the Godunov scheme is tested on different traffic models. The main test is a simplified model of Trondheim, Norway. The results are presented in videos, as well as graphs and tables that show the duration of the driving time through different routes of the model.

Denne artikkelen viser hvordan man kan bruke Lighthill–Witham–Richards modellen og Godunovs metode til å lage et simuleringsverktøy som simulerer trafikkflyten i et nettverk av veier. Først diskuteres konservasjonslover, og hvordan man kan løse disse med karakteristikker. Dette motiverer diskusjonen for svake løsninger. Svake løsninger kan løses ved hjelp av sjokk- og vifte-løsninger. Dette brukes til å beskrive trafikkflyten på en rett vei ved hjelp av LWR-modellen. Deretter gis teori om trafikk i et veikryss, og hvordan man kan finne maksimal fluks gjennom krysset. Forskjellige måter å løse konservasjonslover numerisk blir gitt, og dette gir motivasjon for bruken av Godunovs metode. Ulike typer av trafikkmodeller testes ved hjelp av et program som er basert på LWR-modellen og Godunovs metode. Hovedtesten er en forenklet verson av Trondheim. Trafikkens oppførsel beskrevet av denne modellen blir presentert i korte filmer. Det gis også tabeller og grafer som viser kjøretiden gjennom forskjellige kjøreruter i modellen.

Preface

This thesis is written to fulfill the requirements of the Master of Science degree in Mathematics at Norwegian University of Science and Technology. The assignment was given and supervised by Helge Holden. My paper has been written in the fall semester of 2013 and the spring semester of 2014. The thesis has been made solely by the me, but most of the works has been done by the research of others, and I have done my best to provide references to these sources. With an exception of example 2.3.1 (from [1]), all examples and coding have been done by the me.

One of the harder tasks was to transform how to use the maximizing functions, from the analytical part, into my code. During the process I feel that my programming skills have been improved and that I have got a better understanding of conservation laws and differential equations.

Acknowledgements

I would like to thank Helge Holden for his work as my advisor. I am also grateful to Chris Busuttil who could answer all my questions concerning LaTeX and Thor Mikkel 'Thorvall' Nordahl for proof reading my thesis. Finally I would like to thank Rose-Marie Aker for all her support and patience during this year.

Contents

List of Figures

1 Introduction

The goal of this paper is to create a computer program that simulates road traffic on a network of roads and junctions. This program can be used to look at how traffic will develop after some time in different situations, like a narrowing of a road, traffic lights and roundabouts. The aims of this analysis are principally represented by the maximization of car flow, and the minimization of traffic congestion.

In mathmatics and civil enginering there has been a lot of studies of traffic flow. With different approaches. One possibilty is to model the cars one by one, which is called a microscopic model. Another way is the mesoscopic or kinetic model, where you define a function which expresses the probability of having a vehicle at time t in position x that drives in a given velocty. The last main way to look at the problem is the macroscopic model, which is the one used in the paper. The macroscopic model uses systems of partial differential equations, just like in fluied dynamics.

The model used in this thesis is the one presented by Lighthill, Whitham and Richards in the 1950's. The idea of the model is to describe traffic just like water flow, which can be described in terms of conservation laws:

$$\rho_t + \left[v_{\max}\rho(1 - \frac{\rho}{\rho_{\max}}) \right]_x = 0, \quad x \in \mathbb{R},\ t > 0.$$

A numerical method to solve equations along a road is represented by the Godunov scheme, suggested by S. K. Godunov in 1959. The Godunov scheme is based on exact solutions of the Riemann problem and approximations of piecewise constant functions. All coding has been done in Matlab R2013b.

Chapters 3 and 4 gives the mathematical background on how to solve the trafficking problem with the LWR-model on a single road. First by discussing basic rules about conservation laws, then showing how this can be used to describe traffic flow. Then in chapters 4 and 5 we look at networks of roads and junctions, and how to maximize the fluxes through junctions with different amount of outgoing and incoming roads. In chapters 6 we first give some motivation of why to use the Godunov Scheme, then we explain how the scheme is created. Chapter 7 consists of examples using the LWR-model and the Godunov scheme and the results we get. Then in chapter 8 I give some comments about the results, my models, the code and future work.

2 Conservation Laws

The model for traffic flow in this paper is based on systems of conservation laws, which are a system of partial differential equations, where the variables are conserved quantities, i.e. quantities which can neither be created nor destroyed. In this chapter I give some basic preliminaries about systems of conservation laws.

2.1 Basic Definitions and Assumptions

A system of conservations laws in one space dimension can be written in the form

$$u_t + f(u)_x = 0 \tag{2.1.1}$$

where $u : [0, +\infty) \times \mathbb{R} \to \mathbb{R}^n$ is the "conserved quantity" and $f : \mathbb{R}^n \to \mathbb{R}^n$ is the flux. If we integrate (2.1.1) on an arbitrary interval $[x_1, x_2]$ we get

$$\frac{d}{dt}\int_{x_1}^{x_2} u(t,x)dx = -\int_{x_1}^{x_2} f(u(t,x))_x dx = f(u(t,x_1)) - f(u(t,x_2)),$$

and so the amount of u in $[x_1, x_2]$ varies according to the quantity of u entering at $x = x_1$ and exiting at $x = x_2$.
We always assume f to be smooth. If u is a smooth solution, then (2.1.1) can be written in the quasi linear form

$$u_t + A(u)u_x = 0,$$

where $A(u)$ is the Jacobian matrix of f at u. If $n = 1$, so u takes values in $\mathbb{R}$ and $f : \mathbb{R} \to \mathbb{R}$ then (2.1.1) is a single equation. We then say that (2.1.1) is a scalar equation. In this paper we only deal with the scalar case.

2.2 Mathematical theory for scalar conservation laws

The problem

$$\begin{array}{ll} u_t + f(u)_x = 0 & x \in \mathbb{R}, \quad t > 0, \\ u(0,x) = u_0(x) & x \in \mathbb{R}, \end{array} \tag{2.2.1}$$

for some function $f : \mathbb{R} \to \mathbb{R}$ is called a Cauchy problem. This problem can be solved by the *method of characteristics*. Or as we will see, altleast locally.

Definition 2.2.1. *Let $u : \mathbb{R} \times [0, T) \to \mathbb{R}$ be a classical solution of (2.2.1). The solution χ of the inital-value problem*

$$\chi'(t) = f'(u(\chi(t), t)), \qquad t > 0 \qquad \chi(0) = x_0$$

are called the characteristics of (2.2.1).

The main property of the characteristics is that u is constant along:

$$\frac{d}{dt}u(\chi(t),t)) = u_t(\chi(t),t) + u_x(\chi(t),t)\chi'(t) = 0 \qquad t > 0$$

and hence $u(\chi(t),t)) = const$ for $t > 0$. We illustrate this with an example.

Example 2.2.1. *Let $f(x) = u^2/2$. The characteristics of (2.2.1) is then given by the solutions of*

$$\chi'(t) = u(\chi(t),t), \qquad t > 0 \qquad \chi(0) = x_0.$$

Since $u(\chi(t),t)) = const$ for $t > 0$, χ is a straight line in the (x,t)-plane through x_0 with slope $1/u_0(\chi(t)) = 1/u_0(x_0)$. Characteristics allow to illustrate solutions of (2.2.1) in a compact form.
To show how a solution might look like, let the initial values be:

$$u_0(x) = \begin{cases} 1 & : \quad x < 0 \\ 1 - x & : \quad 0 \leq x < 1 \\ 0 & : \quad x \geq 1 \end{cases} \tag{2.2.2}$$

The characteristics in this case is drawn in Figure 2.2.1.

As we can see in Figure 2.2.1, the characteristics collide at $t = 1$. In other words

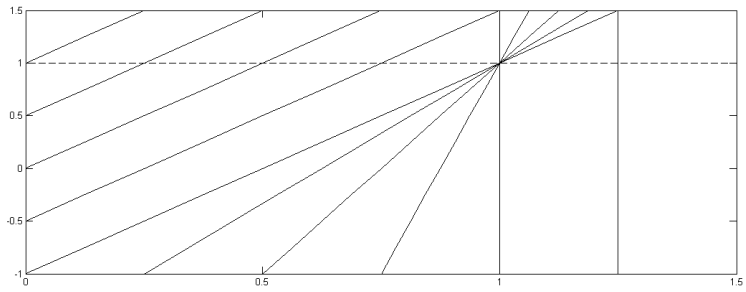

Figure 2.2.1: **Example 2.2.1**. Characteristics of (2.2.1) with (2.2.2) as initial value.

the solutions of (2.2.1) might develop discontinuities in finite time. Therefore we need a solution method including discontinuous functions. Let u be a classical solution to (2.2.1); we multiply this solution by $\phi \in C_o^1(\mathbb{R}^2)$ and integrating over $\mathbb{R}^2$. This gives :

$$0 = \int_0^\infty \int_{\mathbb{R}} (u_t + f(u)_x)\phi dx dt = -\int_0^\infty \int_{\mathbb{R}} (u\phi_t + f(u)\phi_x) dx dt - \int_{\mathbb{R}} u(x,0)\phi(x,0) dx$$

To define the two last integrals we only need a integrable function u. We get the following definition.

Definition 2.2.2. *The function $u : \mathbb{R} \times (0,T) \to \mathbb{R}$ is calleda weak solution of (2.2.1) if for all $\phi \in C_0^1(\mathbb{R}^2)$*

$$\int_0^\infty \int_{\mathbb{R}} (u\phi_t + f(u)\phi_x) dx dt = -\int_{\mathbb{R}} u(x)_0 \phi(x,0) dx.$$

So we need u and $f(u)$ to be integrable, but there are other requirements on u we need for this condition to hold, but we do not specify them. It is possible to check that every classical solution is a weak solution, but not every weak solution needs to be a classical solution.

Now we look at conservation laws with some different types of discontinuous initial values.

Definition 2.2.3. *The problem (2.2.1) with initial values*

$$u_0(x) = \begin{cases} u_l & : \quad x < 0 \\ u_r & : \quad x \leq 0 \end{cases} \tag{2.2.3}$$

and $u_l, u_r \in \mathbb{R}$ is called a Riemann problem.

We want to solve the Riemann problem (2.2.1), (2.2.3). We observe that both $u(x,t)$ and $u(\alpha x, \alpha t)$ is a solution of (2.2.1),(2.2.3) for any $\alpha > 0$. So u only depends on $\xi = x/t$, i.e. $u = u(\xi)$. We have

$$0 = u_t + f(u)_x = -\frac{x}{t^2} u'(\xi) + f'(u(\xi)) u'(\xi) \frac{1}{t} = \frac{1}{t} u'(\xi)(f'(\xi) - \xi)$$

There are several possibilities of what might occur:
$\bullet u'(\xi) = 0 \qquad \Rightarrow \qquad u(\xi) = \text{const}$
$\bullet f'(u(\xi)) = \xi \qquad \Rightarrow \qquad u(\xi) = (f')^{-1}(\xi) = \text{const}$ (if the inverse of f' exists; sufficient condition is $f'' < 0$ or $f'' > 0$ in $\mathbb{R}$)
• u is discontinuous along $\xi = \frac{x}{t}$ i.e. $u'(\xi)$ does not exist.
This motivates us to look at three different cases.

-*Case 1:* $f'(u_l) = f'(u_r)$. This gives the solution $u(x,t) = u_l = u_r$ for all $x \in \mathbb{R}$ and $t > 0$.
-*Case 2:* $f'(u_1) > f'(u_2)$ As the flow direction is from the left to the right, we expect a shock line, i.e. a *discontinuity curve* $x = \psi(t)$. We claim that the discountinuous function

$$u(x,t) = \begin{cases} u_l & : \quad x < st \\ u_r & : \quad x \leq st \end{cases} \tag{2.2.4}$$

is a weak solution to (2.2.1),(2.2.3). Then the discontinuity line is given by $x = \psi(t) = st$ and $s = \psi'(t)$ is the *shock speed* which has to be determined. To prove our claim let $\phi \in C_0^1(\mathbb{R})$. Since $u = \text{const}$ except on $x = st$,

$$\begin{aligned}\int_0^\infty \int_{\mathbb{R}} u\phi_t dxdt &= \int_0^\infty \Big(\int_{-\infty}^{st} u\phi_t dx + \int_{st}^{\infty} u\phi_t dx\Big)dt \\ &= \int_0^\infty \Big(\partial_t \int_{-\infty}^{st} u\phi dx - s\cdot u(st-0,t)\phi(st,t) \\ &\quad + \partial_t \int_{st}^{\infty} u\phi dx + s\cdot u(st+0,t)\phi(st,t)\Big)dt \\ &= -\int_{\mathbb{R}} u(x,0)\phi(x,0)dx - s\cdot(u_l-u_r)\int_0^\infty \phi(st,t)dt\end{aligned}$$

and, by integration by parts,

$$\begin{aligned}\int_0^\infty \int_{\mathbb{R}} f(u)\phi_x dxdt &= \int_0^\infty \Big(-\int_{-\infty}^{st} f(u)_x\phi dx + f(u(st-0,t))\phi(st,t)t \\ &\quad - \int_{st}^{\infty} f(u)_x\phi dx - f(u(st+0,t))\phi(st,t)\Big)dt \\ &= (f(u_l)-f(u_r))\int_0^\infty \phi(st,t)dt.\end{aligned}$$

We conclude

$$\begin{aligned}\int_0^\infty \int_{\mathbb{R}} (u\phi_t + f(u)\phi_x)dxdt &= -\int_{\mathbb{R}} u_0\phi(x,0)dx \\ -[s\cdot(u_l-u_r) - (f(u_l)-f(u_r))]\int_{\mathbb{R}} \phi(st,t)dt.&\end{aligned}$$

Thus

$$s = \frac{f(u_l)-f(u_r)}{u_l-u_r}. \tag{2.2.5}$$

The choice (2.2.5) is called the *Rankine-Hugoniot condition*. The discontinuity curve is always a straight curve for a Riemann problem, i.e., s is always constant. This may not be true for general initial data. In this situation, the Rankine-Hugoniot condition generalizes to

$$s(t) = \psi'(t) = \frac{f(u_l(t))-f(u_r(t))}{u_l(t)-u_r(t)}. \tag{2.2.6}$$

where

$$u_l(t) = \lim_{x\nearrow\psi(t)} u(x,t), \qquad u_r(t) = \lim_{x\searrow\psi(t)} u(x,t). \tag{2.2.7}$$

It can be shown that (2.2.4) is the unique weak solution of (2.2.1), see Theorem 2.3.2.

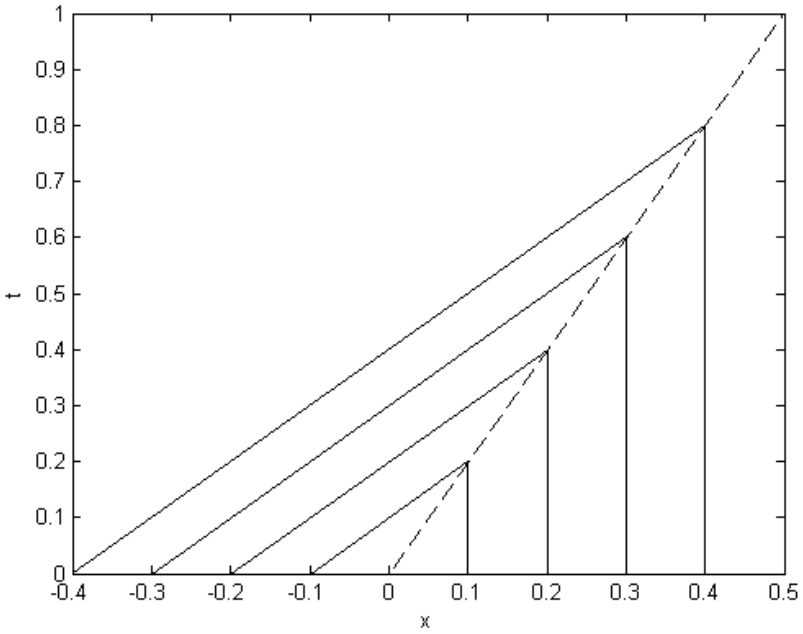

Figure 2.2.2: **Example 2.2.2**. Characteristics of (2.2.1) $f(u) = u(1-u)$, with $u_r = 1/2$ and $u_l = 0$. The dotted line is the shock speed $s = t/2$.

Example 2.2.2. *Let $f(u) = u(1-u)$ and $u_l = 0, u_r = 1/2$. Then the shock speed is $s = 1 - u_l - u_r = \frac{1}{2}$. The solution of (2.2.1),(2.2.3) is illustrated in Figure 2.2.2.*

-*Case 3*: $f(u_l) < f(u_r)$. One solution is given by (2.2.4),

$$u_1(x,t) = \begin{cases} u_l & : \quad x < st \\ u_r & : \quad x \leq st \end{cases} \tag{2.2.8}$$

It is also possible to show that

$$u_2(x,t) = \begin{cases} u_l & : \quad x < f'(u_l)t \\ (f')^{-1}(\frac{x}{t}) & : \quad f'(u_l)t \leq x \leq f'(u_r)t \\ u_r & : \quad x > f'(u_r)t \end{cases} \tag{2.2.9}$$

is a weak solution. In fact, it is possible to show that the problem (2.2.1), (2.2.3) possesses infinitely many weak solutions! What is the physically meaningful solution?
We are going to show that u_2, which is called a *rarefraction wave*, is the physically correct solution. This leads to the notion of *entropy condition*.

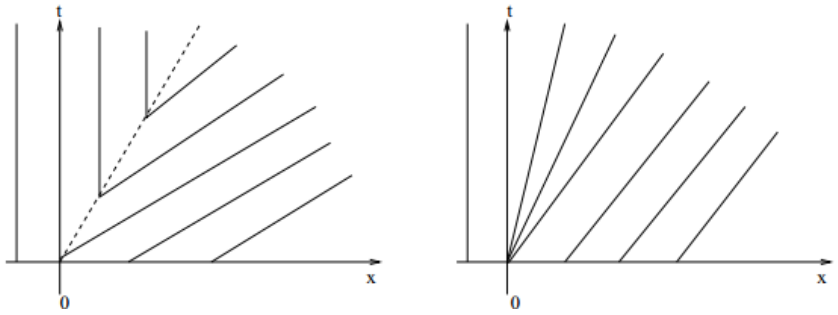

Figure 2.2.3: Characteristics of (2.2.1) and (2.2.3) with $f(u) = u^2/2$ and $u_l = 0$ and $u_r = 1$, corresponding to u_1 (left) and u_2 (right)

.

2.3 Entropy solutions

For $f(u) = u(1-u)$, the condition $u_l > u_r$ in traffic flow interpretation means that there are more cars in $\{x < 0\}$ than in $\{x > 0\}$. The solution of u_1 would mean that all cars to the left of the shock drives with the same velocity, and all cars to the right of the shock drives with the same velocity. Additionally the drivers to the left of the shock drives with a lower velocity. It would be more realistic if the drivers to the left of the shock tried to drive with the same velocity as the drivers to the right of the shock. So the rarefraction solution u_2 would seem like a more physically relevant solution.

Definition 2.3.1. *A weak solution $u : \mathbb{R} \times (0,T) \to \mathbb{R}$ of (2.2.1), (2.2.3) satisfies the entropy condition of Oleinik if and only if along each discontinuity curve $x = \psi(x)$,*

$$\frac{f(u_l(t)) - f(v)}{u_l(t) - v} \leq \psi'(t) \leq \frac{f(u_r(t)) - f(v)}{u_r(t) - v} \tag{2.3.1}$$

for all $t \in (0,T)$ and $u_l(t) < v < u_r(t)$, where $u_l(t)$ and $u_r(t)$ are defined in (2.2.8).

Does u_1 satisfy the entropy condition (2.3.1)? Since

$$\psi'(t) = s(t) = \frac{f(u_r) - f(u_l)}{u_r - u_l}$$

and f assumed to be strictly concave, we obtain for any $u_l < v < u_r$

$$\frac{f(u_l) - f(v)}{u_l - v} < \frac{f(u_l) - f(u_r)}{u_l - u_r} = s < \frac{f(u_r) - f(v)}{u_r - v},$$

which contradicts (2.3.1). Thus u_1 does not satisfy the entropy condition (2.3.1). The function u of Case 2, defined in (2.2.4), however, satisfies (2.3.1) (if f is concave). As the function u_2 is continuous, we do not need to check (2.3.1) for this

function.

A different approach is to use the notion of entropy. We call a function $\eta \in C^2(\mathbb{R})$ an *entropy* and $\psi \in C^1(\mathbb{R})$ an *entropy flux* if and only if η is strictly convex and if for any classical solution u of (2.2.1):

$$\eta(u)_t + \psi(u)_x = 0, \qquad x \in \mathbb{R},\ t > 0 \tag{2.3.2}$$

The idea of this approach is to consider the conservation law as an idealization of a diffusion problem given by the equation

$$u_t + f(u)_x = \epsilon u_{xx} \qquad x \in \mathbb{R},\ t > 0 \tag{2.3.3}$$

where $\epsilon > 0$ is the diffusion coefficient. This equation, together with an initial condition has a unique smooth solution u_ϵ and we assume

$$u_\epsilon \to u \ pointwise\ in\ \mathbb{R} \times (0,T)\ for\ \epsilon \to 0,$$
$$\|\eta'(u_{\epsilon,x})\|_{L^1(\mathbb{R}\times(0,T))} \leq c, \tag{2.3.4}$$

where $c > 0$ is independent of ϵ. The Limit $\epsilon \to 0$ is called the *vanishing viscosity* limit. It can be shown that u is a solution to (2.2.1), and we say that u is the physically relevant solution.
We mulitply (2.3.2) by $\eta'(u_\epsilon)$ and choose $\psi' = f' \cdot \eta'$:

$$\eta(u_\epsilon)_t + \psi(u_\epsilon)_x = \epsilon\eta'(u_\epsilon)u_{\epsilon,xx} = \epsilon(\eta'(u_\epsilon)u_{\epsilon,x})_x - \epsilon\eta''(u_\epsilon)u_{\epsilon,x}^2.$$

Multiplying this equation by $\phi \in C_0^1(\mathbb{R}\times\mathbb{R}), \phi \leq 0$, and integrating over $\mathbb{R} \times (0,\infty)$ gives:

$$\begin{aligned}
\int_0^\infty \int_{\mathbb{R}} (\eta(u_\epsilon)_t + \psi(u_\epsilon)_x)\phi dx dt = \int_0^\infty \Big(& \partial_t \int_{-\infty}^{st} u_1^2 \phi dx - s u_l^2 \phi(st,t) \\
& + \partial_t \int_{st}^{\infty} u_1^2 \phi dx + s u_r^2 \phi(st,t) \Big) \\
& - \epsilon \int_0^\infty \int_{\mathbb{R}} \eta'(u_\epsilon) u_{\epsilon,x} \phi_x dx dt \\
& - \epsilon \int_0^\infty \int_{\mathbb{R}} \eta''(u_\epsilon) u_{\epsilon,x}^2 \phi dx dt \\
& \leq \epsilon \|\eta'(u_\epsilon)u_{\epsilon,x}\|_{L^1(\mathbb{R}\times(0,\infty))} \|\phi_x\|_{L^\infty(\mathbb{R}\times(0,\infty))} \\
& \to 0 \qquad (as\ \epsilon \to 0),
\end{aligned}$$

since $\eta''(u_\epsilon) > 0$ and (2.3.4). As ϕ is arbitrary, we deduce the entropy inequality

$$\eta(u)_t + \psi(u)_x \leq 0. \tag{2.3.5}$$

This inequality only holds for smooth solutions. From the definition of weak solutions it follows that we can write the entropy inequality for weak solutions as

$$\int_0^\infty \int_{\mathbb{R}} (\eta(u)\phi_t + \psi(u)\phi_x) dx dt \leq -\int_{\mathbb{R}} \eta(u_0(x))\phi(x,0) dx \ \ \forall \phi \in C_0^1(\mathbb{R}^2). \tag{2.3.6}$$

Definition 2.3.2. *Let $u : \mathbb{R} \times (0,T) \to \mathbb{R}$ be a weak solution of (2.2.1). Then u is called an entropy solution if and only if for all convex entropies η and corresponding entropy flux ψ, the inequality (2.3.6) holds.*

The function u_2 satisfies the entropy equation (2.3.6) almost everywhere since u_2 is continuous and we can define the derivatives in a weak sense (it is also possible to prove that (2.3.2) is in the weak form similarly to (2.3.6)). So u_2 is an entropy solution. Does this hold for u_1? We show with an example that it does not.

Example 2.3.1. *Let $f(u) = u^2/2$, $\eta(u) = u^2$ so $\psi(u) = \frac{2}{3}u^3$, and let $\phi \in C_0^1(\mathbb{R}^2)$. $\phi \leq 0$. Then, since $s = \frac{1}{2}(u_l + u_r)$,*

$$\begin{aligned}
\int_0^\infty \int_{\mathbb{R}} (u_1^2\phi_t + \frac{2}{3}u_1^3\phi_x)dxdt &= \int_0^\infty \Big(\partial_t \int_{-\infty}^{st} u_1^2\phi dx - su_l^2\phi(st,t) + \partial_t \int_{st}^\infty u_1^2\phi dx \\
&\quad + su_r^2\phi(st,t) + \frac{2}{3}u_l^3\phi(st,t) - \frac{2}{3}u_r^3\phi(st,t)\Big)dt \\
&= -\int_{\mathbb{R}} u_0(x)^2\phi(x,0)dx \\
&\quad - \frac{1}{2}(u_l + u_r)(u_l^2 - u_r^2)\int_0^\infty \phi(st,t)dt \\
&\quad + \frac{2}{3}(u_l^3 - u_r^3)\int_0^\infty \phi(st,t)dt \\
&= -\int_{\mathbb{R}} u_0(x)^2\phi(x,0)dx + \frac{1}{6}(u_l - u_r)^3\int_0^\infty \phi(st,t)dt \\
&\geq -\int_{\mathbb{R}} \eta(u_0(x))\phi(x,0)dx
\end{aligned}$$

if and only if $u_l \geq u_r$. Hence, u_1 is not an entropy solution.

Example 2.3.1 shows that the two equations

$$u_t + \left(\frac{u^2}{2}\right)_x = 0 \qquad \text{and} \qquad (u^2)_t + \frac{2}{3}(u^3)_x = 0$$

are only equivalent for classical solutions.
The above calculations motivates that only the rarefraction wave is the relevant solutions for the Riemann problem if $f'(u_l) > f'(u_r)$. For $f'(u_l) < f'(u_r)$ we have to expect discontinuous solutions with shocks. We summarize the above results in a theorem.

For strictly concave functions we have:

Theorem 2.3.1. *Let f in $C^2(\mathbb{R})$ with $f'' < 0$ in $\mathbb{R}$.*

(1) Let $u_l < u_r$ and set $s = \dfrac{f(u_l) - f(u_r)}{u_l - u_r}$. Then

$$u(x,t) = \begin{cases} u_l & : \quad x < st \\ u_r & : \quad x > st \end{cases}$$

is a weak solution to (2.2.1),(2.2.3) satisfying the entropy condition of Oleinik.

(2) Let $u_l > u_r$. Then

$$u(x,t) = \begin{cases} u_l & : \quad x < f'(u_l)t \\ (f')^{-1}(\frac{x}{t}) & : \quad f'(u_l)t \leq x \leq f'(u_r)t \\ u_r & : \quad x > f'(u_r)t \end{cases}$$

is a weak solution of (2.2.1),(2.2.3).

Notice that for strictly convex functions ($f'' > 0$) $u_l < u_r$ gives a shock solution and $u_l > u_r$ gives a rarefraction solution.
We have written that the problem (2.2.1),(2.2.3) ($f'' < 0$) with $u_l < u_r$ has infinitely many solutions and that the solution u_1 does not satisfy the entropy condition of Oleinik nor is an entropy solution. However, is u_2 the only solution satisfying the entropy condition of Oleinik and condition (2.3.6)? The answer is yes, but not so easy to prove. We only state the result. (see [1]):

Theorem 2.3.2. *Let $f \in C^\infty(\mathbb{R})$ and $u_0 \in L^\infty(\mathbb{R})$. Then there exist at most one entropy solution of (2.2.1) satisfying the entropy condition (2.3.6) holds.*

3 Traffic flow models

Consider the traffic flow on a road with one lane. We want to look at the cars as a density $\rho(x,t)$, say vehicles per kilometer in $x \in \mathbb{R}$ and time $t \geq 0$. The number of the cars on the interval (x_1, x_2) at time t is

$$\int_{x_1}^{x_2} \rho(x,t)dx.$$

Let $v(x,t)$ denote the velocity of the cars at time t. The number of cars which pass through x at time t (in unit length) is $\rho(x,t)v(x,t)$. We need an equation which shows the evolution of the car density. The number of cars in the interval (x_1, x_2) changes according to the number of cars that exits or drives into the interval.

$$\frac{d}{dt}\int_{x_1}^{x_2} \rho(x,t)dx = \rho(x_1,t)v(x_1,t) - \rho(x_2,t)v(x_2,t).$$

Integrating this equation with respect to time and assuming that ρ and v are regular functions yields

$$\begin{aligned}\int_{t_1}^{t_2}\int_{x_1}^{x_2} \partial_t\rho(x,t)dxdt &= \int_{t_1}^{t_2} (\rho(x_1,t)v(x_1,t) - \rho(x_2,t)v(x_2,t))dxdt \\ &= -\int_{t_1}^{t_2}\int_{x_1}^{x_2} \partial_x(\rho(x,t)v(x,t))dxdt.\end{aligned}$$

Since $x_1, x_2 \in \mathbb{R}$, $t_1, t_2 > 0$ are arbitrary, we can conclude

$$\rho_t + (\rho v)_x = 0, \quad x \in \mathbb{R},\ t > 0. \tag{3.0.7}$$

This equation has the same form as the conservation laws described in the previous chapter. We have to add some initial conditions

$$\rho(x,0) = \rho_0(x), \quad x \in \mathbb{R}.$$

We now need an equation for the velocity v. A simple assumption is to assume that the speed of the cars only depends on the density of the cars. If the road is empty, $\rho = 0$, we will drive with maximal velocity $v = v_{\max}$. In heavy traffic we will have to slow down and stop ($v = 0$) in a traffic jam, and the cars will be bumper to bumper ($\rho = \rho_{\max}$). The simplest model is the linear relation

$$v(\rho) = v_{\max}\left(1 - \frac{\rho}{\rho_{\max}}\right), \quad 0 \leq \rho \leq \rho_{\max}.$$

Equation (3.0.7) then becomes

$$\rho_t + \left[v_{\max}\rho(1 - \frac{\rho}{\rho_{\max}})\right]_x = 0, \quad x \in \mathbb{R},\ t > 0$$

This equation is a conservation law since it expresses the conservation of the number of cars. Integrating the equation over $x \in \mathbb{R}$ gives

$$\frac{d}{dt}\int_{\mathbb{R}} \rho(x,t)dx = -\int_{\mathbb{R}} \frac{\partial}{\partial x}\left[v_{\max}(x,t)\left(1-\frac{\rho(x,t)}{\rho_{\max}}\right)\right]dx = 0$$

and we see that the number of cars in $\mathbb{R}$ is constant for all $t \leq 0$.
We will now look at some different traffic models. The first one is the one we just presented.

(1) Lighthill–Whitham–Richards model:

$$\rho_t = (\rho v(\rho))_x = 0, \quad v(\rho) = v_{\max}\left(1-\frac{\rho}{\rho_{\max}}\right), \quad 0 \leq \rho \leq \rho_{\max}.$$

If we set $v_{\max} = 1$ and $\rho_{\max} = 1$ the equation reduces to

$$\rho_t = (\rho(1-\rho))_x = 0.$$

For this model $f = \rho v$ is a C^2 function, f is strictly concave ($f'' = -2$) and $f(0) = f(1) = 0$.
This method of modeling traffic flow originated under the assumption that traffic streams as a whole are comparable to fluid streams. The major first step in macroscopic modeling was taken by Lightill and Whitham in 1955, when they compared 'traffic flow on long crowded roads' with 'flood movement in long rivers'. A year later, Richard complemented the idea with the introduction of 'shock waves on the highway', completing the so-called LWR model.

(2) Greenberg Model:

In this model it is assumed that the velocity of the vehicles can be very large for low densities:

$$\rho_t + (\rho v(\rho))_x, \quad v(\rho) = v_{\max}\ln\left(\frac{\rho_{\max}}{\rho}\right), \quad 0 < \rho \leq \rho_{\max}.$$

In this case $v(\rho_{\max}) = 0$, while v is unbounded when $\rho \to 0^+$.

(3) Payne-Whitham model:

$$\rho_t + (v\rho)_x = 0, \quad (\rho v)_t + (\rho v^2 + p(\rho))_x = 0.$$

This model mimics the flow of gas particles. The above equations are known as the Euler equations of gas dynamics with pressure $p(\rho) = a\rho^\gamma$, $a > 0$, $\gamma \leq 1$. The disadvantage of this model is that there may be solutions for which the velocity v is negative.

(4) Aw-Rascale model:

$$\rho_t + (\rho v)_x = 0, \quad (\rho v + \rho p(\rho))_t + \rho v p(\rho))_x = 0.$$

This model has been proposed as an improvement of the Payne-Witham model and has been derived from microscopic models. $p = p(\rho)$ is the "pressure", an increasing function of the density.

4 Networks

Now we know how to describe traffic flow on a single road. The next goal is to look at what happens when roads meet at a junction. In this chapter we will show how we can represent a traffic network as a direct graph. A directed graph is a collection of direct edges, connected together with some vertices. Each vertex is given by a finite number of incoming and outgoing edges, just as each junction is given by a finite number of incoming and outgoing roads. We will first describe a network of directed graphs, by determining the behavior at vertices. Then we will translate this information to roads and junctions.

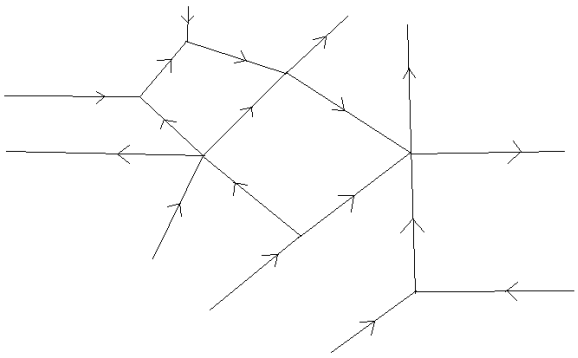

Figure 4.0.1: Example of a network

4.1 Basic Definitions and Assumptions

We begin to state what it means for a function to be of bounded variation, and then we give the definition of a network. For an interval $I \in \mathbb{R}$ and a function $g : I \to \mathbb{R}$. The total variation of g is defined by

$$Tot.Var.g = \sup\{\sum_{j=1}^{N} |g(x_j) - g(x_{j-1})|\},$$

for $N \geq 1$, all the ponits x_j, $j \in \{1, ..., N\}$ belongs to the interval I and are such that $x_0 < x_1 < \cdots < x_N$.

Definition 4.1.1. *A function $g : I \to \mathbb{R}$ has bounded total variation if $Tot.Var.g < +\infty$.*

A function with bounded variation has at most countably many points of discontinuity and these functions does not occilate vigorously. (See [4, pp. 281-284]).

Now we can start to talk about networks.

Definition 4.1.2. *A network is a couple $(\mathcal{I}, \mathcal{J})$ where*
$\mathcal{I}$ is a finite collection of edges, which are intervals in $\mathbb{R}$, $I_i = [a_i, b_i] \subseteq \mathbb{R}$, i=1,...,N;
$\mathcal{J}$ is a finite collection of vertices. Each vertex J is a union of two nonempty subsets $Inc(J)$ and $Out(J)$ of $\{1,...,N\}$.

We assume the following:

1. For every $J \neq J' \in \mathcal{J}$ we have $Inc(J) \cap Inc(J') = \emptyset$ and $Out(J) \cap Out(J') = \emptyset$.
2. If $i \notin \cup_{J \in \mathcal{J}} Inc(J)$ then $b_i = +\infty$ and if $i \notin \cup_{J \in \mathcal{J}} Out(J)$ then $a_i = -\infty$. Moreover, the two cases are not mutually exclusive.

This is just saying that the network is a graph. Each vertex can be numbered, and represented as a $n+m$-tuple $(i_1, ..., i_n, i_{n+1}, ..., i_{n+m})$. The n is number of incoming edges and m is the number of outgoing edges. The first condition states that each edge can be incoming of at just one vertex, and outgoing for just one vertex, and that they are connected to at least one vertex. See Figure 4.0.1.

4.2 Riemann Solvers

In this section we assume that the traffic on each edge is represented by an hyperbolic system of conservation laws:

$$(u_i)_t + (f_i(u_i))_x = 0 \quad u_i \in \mathbb{R}^p, \tag{4.2.1}$$

the goal is to define and solve Riemann problems at vertices. Given network $(\mathcal{I}, \mathcal{J})$ and a vertex $J \in \mathcal{J}$ and assume that $\text{Inc}(J) = \{1, ..., n\}$ and $\text{Out}(J) = \{n+1, ..., n+m\}$.

Definition 4.2.1. *A Riemann problem at J is a Cauchy problem corresponding to an initial value which is constant on each edge.*

Since by the definition of the Riemann problem for J, we have constant initial value on each edge, we need only to look for centered solutions. In other words, for a Riemann problem shocks, rarefractions or contact discontinuities will be formed at every edge.

Definition 4.2.2. *A Riemann solver for the vertex J is a function*

$$RS : (\mathbb{R}^p)^{n+m} \to (\mathbb{R}^p)^{n+m}$$

that associates to every Riemann data $u_0 = (u_{1,0}, ..., u_{n+m,0})$ at J a vector $\hat{u} = (\hat{u}_1, ...\hat{u}_{n+m})$ so that the following holds.
On each edge $I_i, i = 1, ..., n+m$, the solution is given by the solution to the initial-boundary value problem with initial value $u_{i,0}$ and the boundary data $\hat{u}_i$. We require the consistency condition

(CC) $RS(RS(u_0)) = RS(u_0)$.

Now we can define admissible solutions at J.

Definition 4.2.3. *Assume a Riemann Solver RS is assigned at a junction J. Let $u = (u_1, ..., u_{n+m})$, $u_i : [0, +\infty) \times I_i \to \mathbb{R}^p$ be such that $u_i(t, \cdot)$ is of bounded variation for every $t \geq 0$. Then u is an admissible weak solution to (4.2.1) related to RS at the vertex J if and only if the following properties hold:*

(i) u_i is a weak solution to (4.2.1) on the edge;
(ii) for almost every $t > 0$, setting

$$u_J(t) = (u_1(\cdot, b_1-), ..., u_n(\cdot, b_n-), u_{n+1}(\cdot, a_{n+1}+), ..., u_{n+m}(\cdot, a_{n+m}+)),$$

we have

$$RS(u_J(t)) = u_J(t).$$

in traffic modeling cars can not disappear or be created at a junction. So for traffic the quantity u must be conserved at the vertex J. In other words the total flux in to the junction must be the same as the total flux out of the junction. Therefore, necessary condition is to ask equality of incoming and outgoing fluxes for the obtained solution of $\hat{u}$. But this is not enough, because the initial-boundary value problem on each edge may produce a solution which does not attain the boundary value pointwise. To ensure conservation of u, we need that the solutions to the initial boundary value problem have negative characteristic velocities on incoming edges and positive characteristic velocities on outgoing ones. This adds up to ask the Riemann problem on the real line with initial data $(u_{i,0}, \hat{u}_i)$, $i = 1, ..., n + m$, to only produces waves with positive velocities. Conservation of u at the vertex J is the same as to ask:

Cons.1 if $\hat{u} = RS(u_0)$, then for incoming edges the solution to the Riemann problem $(u_{i,0}, \hat{u}_i)$ admits waves with strictly negative speed, $i = 1, ..., n$, while for outgoing edges the solution to the Riemann problem $(\hat{u}_j, u_{j,0})$ admits all waves with positive speed, $j = n + 1, ..., n + m$.
Cons.2 if $\hat{u} = RS(u_0)$, then the incoming flux is equal to the outgoing one, i.e.:

$$\sum_{i=1}^{n} f_i(\hat{u}_i) = \sum_{j=n+1}^{n+m} f_j(\hat{u}_j).$$

These to conditions gives that the sum of traces of fluxes over incoming edges is equal to the sum of traces of fluxes over the outgoing edges.

In order to find out how much traffic that can flow though a junction, i.e. images of which region the Riemann solver belongs to, we need some preliminary results:
(F) $f : [0, 1] \to \mathbb{R}$ is a smooth, strictly concave and satisfies $f(0) = f(1) = 0$. f a unique maximum $\sigma \in (0, 1)$ such that $f'(\sigma) = 0$, i.e σ is a strict maximum.

Definition 4.2.4. *Let $\tau : [0,1] \to [0,1]$ be a map such that:*
1. $f(\tau(u)) = f(u)$ for every $u \in [0,1]$;
2. $\tau(u) \neq u$ for every $u \in [0,1] \backslash \{\sigma\}$.

Proposition 4.2.1. *The function τ is well defined and continuoues. Moreover it satisfies*

$$0 \leq u \leq \sigma \iff \sigma \leq \tau(u) \leq 1, \quad \sigma \leq u \leq 1 \iff 0 \leq \tau(u) \leq \sigma.$$

Proof. Fix $u \in [0,1]$. If $u = \sigma$, then $\tau(u) = u$, since there is just one point of maximum for f. If $u \neq \sigma$, then by 1., $\tau(u)$ can assume at most two values. One is u itself, while the other belongs to $(\sigma, 1]$ if $u < \sigma$ or it belongs to $[0, \sigma)$ if $u > \sigma$. Since we want that $\tau(u) \neq u$ if $u \neq \sigma$, then τ is clearly well defined and the equation is satisfied.
The continuity of τ follows from the regularity of the flux f. □

Now we have what we need to construct all regions for which the images of all possible Riemann solvers exists. This is described by the next proposition.

Proposition 4.2.2. *Fix a vertex J, an initial value $(u_{1,0}, ..., u_{n+m,0})$ and a Riemann solver RS satisfying **Cons.1** and **Cons.2**. Define*

$$(\hat{u}_1, ..., \hat{u}_{n+m}) = RS(u_{1,0}, ..., u_{n+m,0}).$$

For an incoming edge I_i the following possibilities hold :
1. if the initial value $u_{i,0} \in [0, \sigma]$, then

$$\hat{u}_i \in \{u_{i,0}\} \cup (\tau(u_{i,0}), 1];$$

2. if the initial value $u_{i,0} \in [\sigma, 1]$, then

$$\hat{u}_i \in [\sigma, 1].$$

For an outgoing edge I_j the following possibilities hold:
1. if the initial value $u_{j,0} \in [0, \sigma]$, then

$$\hat{u}_j \in [0, \sigma];$$

2. if the initial value $u_{j,0} \in [\sigma, 1]$, then

$$\hat{u}_j \in \{u_{j,0}\} \cup [0, \tau(u_{j,0})).$$

The proof of this proposition can be found in [2, pp. 101-102]. The proposition allows us to introduce the following functions. For each incoming edge I_i, define

$$\gamma_i^{\max}(u_{i,0}) = \begin{cases} f(u_{i,0}), & \text{if} \quad u_{i,0} \in [0, \sigma] \\ f(\sigma) & \text{if} \quad u_{i,0} \in (\sigma, 1] \end{cases} \tag{4.2.2}$$

while for each outgoing edge I_j, define

$$\gamma_j^{\max}(u_{j,0}) = \begin{cases} f(\sigma), & \text{if} \quad u_{j,0} \in [0, \sigma] \\ f(u_{j,0}) & \text{if} \quad u_{j,0} \in (\sigma, 1] \end{cases} \tag{4.2.3}$$

The quantities $\gamma_i^{\max}(u_{i,0})$ and $\gamma_j^{\max}(u_{j,0})$ represent the maximum flux that can be obtained by a single wave solution on each road.

5 Lighthill–Witham–Richards Model on Networks

The LWR-model will be used on each road and at each junction. We look at Riemann solvers that satisfy the conservation of cars (Con1 and Con2) and the following rules:

(A) In some way drivers choose where to drive, that is the traffic from incoming roads is distributed on outgoing roads according to fixed coefficients;
(B) Respecting (A), the drivers choose to maximize fluxes.

When there are more incoming roads than outgoing we need a right of way parameter that describes how many cars that can drive through the junction from the incoming roads.

5.1 Basic Defintions and Assumptions

Definition 5.1.1. *A road network is a network. The edges is represented by undirectional roads and the vertices by junctions.*

Given a road network $(\mathcal{I}, \mathcal{J})$. On each road we have the equation

$$\rho_t + f(\rho)_x = 0, \tag{5.1.1}$$

where $\rho = \rho(x,t) \in [0, \rho_{\max}]$, $(t,x) \in \mathbb{R}_+ \times \mathbb{R}$, is the density of cars, v is the average speed and $f(\rho) = v\rho$ is the flux. We assume the following:
(A1) $\rho_{\max} = 1$;
(A2) the speed v depends only on the density ρ;
(A3) the flux f is a strictly concave C^2 function;
(A4) $f(0) = f(1) = 0$.

(A3) and (A4) gives that f has a unique point of maximum $\sigma \in (0,1)$.

To distribute the traffic at each junction, we give each junction J a traffic-distribution matrix, i.e a matrix describing the percentage of cars from outgoing to incoming roads.

Definition 5.1.2. *Given a junction J with n incoming roads, say $I_1, .., I_n$, and m outgoing roads, say $I_{n+1}, ..., I_{n+m}$. Then, the traffic distribution matrix A is given by*

$$A = \begin{pmatrix} \alpha_{n+1,1} & \cdots & \alpha_{n+1,n} \\ \vdots & \vdots & \vdots \\ \alpha_{n+m,1} & \cdots & \alpha_{m+n,n} \end{pmatrix} \tag{5.1.2}$$

where $0 \leq \alpha_{i,j} \leq 1$ for all $i \in \{1, ..., n\}$ and all $j \in \{n+1, ..., n+m\}$ and

$$\sum_{j=n+1}^{n+m} \alpha_{i,j} = 1 \tag{5.1.3}$$

for every $i \in \{1, ..., n\}$.

Given a junction J and an incoming road I_i, the i-th column of A describes how the traffic from I_i distributes in percentages to the outgoing roads. So if C is the quantity of traffic coming from road I_i then $\alpha_{i,j}C$ traffic moves toward roads I_j.
We introduce a technical condition on the matrix A. We say that the matrix A satisfies hypothesis (C) if the following holds.

(C) Let $\{e_1, ..., e_n\}$ be the canonical basis of $\mathbb{R}^n$ and for every subset $V \subset \mathbb{R}^n$ indicated by $V^\perp$ its orthogonal. Define for every $i = 1, .., n$, $H_i = \{e_i\}^\perp$, i.e the coordinate hyperplane orthogonal to e_i and, for every $j = n+1, ..., n+m$ let, $\alpha_j = (\alpha_{j1}, ..., \alpha_{jn}) \in \mathbb{R}^n$ and define $H_j = \{\alpha_j\}^\perp$. Let K be the set of indices $k = (k_1, ..., k_l)$, $1 \leq l \leq n-1$, such that $0 \leq k_1 < k_2 < \cdots < k_l \leq n+m$ and for every $k \in K$ set $H_k = \bigcap_{h=1}^{l} H_{k_h}$. Letting $\mathbf{1} = (1, ..., 1) \in \mathbb{R}^n$, then for every $k \in K$,

$$\mathbf{1} \notin H_k^\perp \tag{5.1.4}$$

Condition (C) is a technical condition, which is important to isolate the unique solution to Riemann problems at a junction. From (C) we see that $m \geq n$. If that is not the case, then by the definiton $\mathbf{1} = \sum_{j=n+1}^{n+m} \alpha_j$, we get $\mathbf{1} \in H_k^\perp$, where

$$H_k = \cup_{j=n+1}^{n+m} H_j.$$

The case where $m = n$ we can check that condition (C) is generic in the space of $n \times n$ matrices, which means that the set of matrices satisfying (C) is open and dense.
If $n \geq 2$, then (C) gives that, for every $j \in \{n+1, ..., n+m\}$ and for every distinct elements $i, i' \in \{1, ..., n\}$, $\alpha_{j,i} \neq \alpha_{j,i'}$ holds. Otherwise, without loss of generality, we may suppose that $\alpha_{n+1,1} = \alpha_{n+1,2}$. Since

$$H = (\cap_{2<j\leq n} H_j) \cap H_{n+1},$$

then, by condition (C), there exists an element $(x_1, x_2, 0, ..., 0) \in H$ such that $x_1 + x_2 \neq 0$ and $\alpha_{n+1,1}(x_1 + x_2) = 0$
In the case of a simple junction J with 2 incoming roads and 2 outgoing roads, then (C) is equivalent to that, for all $j \in 3, 4$, $\alpha_{j,1} \neq \alpha_{j,2}$.
From here on we will assume that each traffic-distribution matrix satisfies hypothesis (C).

We write $\rho_i : [0, +\infty) \times I_i \to [0, 1]$ for the density of cars in the road I_i of the network. We want ρ_i to be a weak entropic solution on I_i, i.e. for every smooth function $\varphi : [0, +\infty) \times I_i \to \mathbb{R}$ with compact support on $(0, +\infty) \times (a_i, b_i)$,

$$\int_0^{+\infty} \int_{a_i}^{b_i} \left(\rho_i \frac{\partial \varphi}{\partial t} + f(\rho_i) \frac{\partial \varphi}{\partial x} \right) dxdt = 0, \tag{5.1.5}$$

and for every $k \in \mathbb{R}$ and every smooth $\tilde{\varphi} : [0, +\infty) \times I_i \to \mathbb{R}$, positive with compact support on $(0, +\infty) \times (a_i, b_i)$,

$$\int_0^{+\infty} \int_{a_i}^{b_i} \left(|\rho_i - k| \frac{\partial \tilde{\varphi}}{\partial t} + sgn(\rho_i - k)(f(\rho_i - f(k)) \frac{\partial \tilde{\varphi}}{\partial x} \right) dxdt \geq 0. \tag{5.1.6}$$

Definition 5.1.3. *Let J be a junction with incoming roads, say $I_1, ..., I_n$, and outgoing roads, say $I_{n+1}, ..., I_{n+m}$. A weak solution at J is a collection of functions $\rho_l : [0, +\infty) \times I_l \to \mathbb{R}$, $l = 1, .., n+m$, such that*

$$\sum_{l=1}^{n+m} \left(\int_0^{+\infty} \int_{a_l}^{b_l} \left(\rho_l \frac{\partial \varphi_l}{\partial t} + f(\rho_l) \frac{\partial \varphi_l}{\partial x} \right) dxdt \right) = 0, \tag{5.1.7}$$

for every smooth φ, $l = 1, ..., n+m$, having compact support in the set $(0, \infty) \times (a_l, b_l]$ for $l = 1, .., n$ (incoming roads) and in $(0, \infty) \times [a_l, b_l)$ for $l = n+1, ..., n+m$ (outgoing roads), that are also smooth across the junction, i.e.

$$\varphi_i(\cdot, b_i) = \varphi_j(\cdot, a_j) \qquad \frac{\partial \varphi_i}{\partial x}(\cdot, b_i) = \frac{\partial \varphi_j}{\partial x}(\cdot, a_j)$$

where $i \in \{1, .., n\}$ and $j \in \{n+1, .., n+m\}$.

Lemma 5.1.1. *Let $\rho = (\rho_1, ..., \rho_{n+m})$ be a weak solution at the junction such that each $x \to \rho_i(t, x)$ has bounded variation. Then ρ satisfies the Rankine-Hugonit Condition at the Junction J, namely*

$$\sum_{i=1}^{n} f(\rho_i(t, b_i-)) = \sum_{j=n+1}^{n+m} f(\rho_j(t, a_j+)), \tag{5.1.8}$$

for almost every $t > 0$.

Proof. Suppose for simplicity that, for every $l \in \{1, ..., n+m\}$, ρ_l is constant on I_l. Then (5.1.7) implies that

$$\sum_{l=1}^{n+m} \int_0^{+\infty} \int_{a_l}^{b_l} div(\rho_l \varphi_l, f(\rho_l)\varphi_l) dxdt = 0.$$

By the divergence theorem to the last expression and by using the hypothesis on the function φ_l we get

$$\int_0^{+\infty} \left(\sum_{l=1}^{n} f(\rho_l(t, b_l)) - \sum_{j=n+1}^{n+m} f(\rho_l(t, a_l)) \right) \varphi_1(t, b_l) dt = 0$$

and so

$$\sum_{l=1}^{n} f(\rho_l(t, b_l)) = \sum_{l=n+1}^{n+m} f(\rho_l(t, a_l))$$

by the arbitrariness of the function φ_1. □

Definition 5.1.4. *Let $\rho = (\rho_1, ..., \rho_{n+m})$ be such that $\rho_i(t, \cdot)$ is of bounded variation for every $t \geq 0$. Then ρ is an admissible weak solution of (5.1.1) related to matrix A at the junction J if and only if the following properties hold:*

(i) ρ is a weak solution at the junction J;
(ii) $f(\rho_j(\cdot, a_j+)) = \sum_{i=1}^{n} \alpha_{j,i} f(\rho_i(\cdot, b_i-))$, for each $j = n+1, ..., n+m$;
(iii) $\sum_{i=1}^{n} f(\rho_i(\cdot, b_i))$ is maximum subject to (i) and (ii).

(i) is equivalent the conservation of cars at junctions. (ii) and (iii) describes the rules (A) and (B), the preferences of drivers and the maximization procedure.

Definition 5.1.5. *Given $\bar{\rho}_i : I_i \to \mathbb{R}$, $i = 1, ..., N$, L^∞ functions, a collection of functions $\rho = (\rho_1, ..., \rho_N)$, with $\rho_i : [0, +\infty) \times I_i \to \mathbb{R}$ continous function from $[0, +\infty)$ into L^1_{loc}, is an admissible solution if ρ_i is a weak entropic solution to (5.1.1) on I_i, $\rho_i(0, x) = \bar{\rho}_i(x)$ a.e., at each junction ρ is a weak solution and is an admissible weak solution in the case of bounded variation.*

For all the roads $I_i = [a_i, b_i]$ of a network. If $a_i > -\infty$, we assume, by definition 4.1.2 that it is an incoming road of a junction. Likewise, if $b_i < +\infty$ it is an outgoing road of a junction. Then a solution for every time is determined just by initial value on the network.
In the real world we have no infinite roads. So if a road has a_i finite but is not outgoing we have to assign boundary data. Likwise if b_i finite, but not incoming for any junction.

5.2 The Riemann Problem at Junctions

The next step is to construct a Riemann solver at junctions, satisfying rules (A) and (B). We will look at the case of a junction where there are more incoming roads than outgoing, and the case where there are less incoming than outgoing. We go in special detail in the case where there are one incoming road and two outgoing.

For a junction J with n incoming roads and m outgoing roads (Figure 5.2.1) and a distribution matrix A. We indicate by

$$(t, x) \in \mathbb{R}_+ \times I_i \to \rho_i(t, x) \in [0, 1], \quad i = 1, ..., n \tag{5.2.1}$$

the densities of cars on the roads with incoming traffic and

$$(t, x) \in \mathbb{R}_+ \times I_j \to \rho_j(t, x) \in [0, 1], \quad j = n + 1, ..., n + m \tag{5.2.2}$$

those roads with outgoing traffic. The initial densities are $(\rho_{1,0}, ..., \rho_{n+m,0})$ in each road of the junction J. In this section, we use the function τ of Definition 4.2.4.

The case $n \leq m$

For a junction J, we have more outgoing roads or the same amount as the number of incoming roads.

Theorem 5.2.1. *Consider a junction J, assume (A1)-(A4) and that the matrix A satisfies the condition (C). For every $\rho_{1,0}, ..., \rho_{n+m,0} \in [0, 1]$, there exists a unique admissible centered weak solution $\rho = (\rho_1, ..., \rho_{n+m})$ to (5.1.1) at the junction J, in the sense of Definition 5.1.1, such that*

$$\rho_1(0, \cdot) \equiv \rho_{1,0}, ..., \rho_{n+m}(0, \cdot) \equiv \rho_{n+m,0}.$$

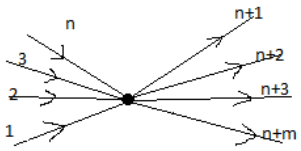

Figure 5.2.1: A junction with n incoming roads and m outgoing roads.

Then there exists a unique $n+m$-tuple $(\hat{\rho}_1, ..., \hat{\rho}_{n+m}) \in [0,1]^{n+m}$ such that

$$\hat{\rho}_i \in \begin{cases} \{\rho_{i,0}\} \cup (\tau(\rho_i), 1], & \text{if} \quad 0 \leq \rho_{i,0} \leq \sigma \\ (\sigma, 1], & \text{if} \quad \sigma \leq \rho_{i,0} \leq 1 \end{cases} \quad i = 1, ..., n \tag{5.2.3}$$

and

$$\hat{\rho}_j \in \begin{cases} [0, \sigma], & \text{if} \quad 0 \leq \rho_{j,0} \leq \sigma \\ \{\rho_{i,0}\} \cup [0, \tau(\rho_j)), & \text{if} \quad \sigma \leq \rho_{i,0} \leq 1 \end{cases} \quad j = n+1, ..., n+m \tag{5.2.4}$$

and for $i \in \{1, ..., n\}$ the solution is given by the wave $(\rho_{i,0}, \hat{\rho}_i)$, while for $j \in \{n+1, ..., n+m\}$ the solution is given by the wave $(\hat{\rho}_j, \rho_{j,0})$.

The above theorem produces the unique Riemann solver RS that gives us an admissible weak solution to the Riemann problem at a junction that satisfies rules (A) and (B).
The theorem leads to the following corrolary.

Corollary 5.2.1. *Consider a junction J, assume (A1)-(A4) and that the matrix A satisfies (C). Then there exist a unique Riemann solver compatible with Definition 5.1.4. Moreover, for every $\rho_{1,0}, ..., \rho_{n+m,0} \in [0,1]$, the $n+m$-tuple $(\hat{\rho}_1, ..., \hat{\rho}_{n+m}) = RS(\rho_{1,0}, ..., \rho_{n+m,0})$ satisfies (5.2.3) and (5.2.4).*

Proof of Theorem 5.2.1. Define the map

$$E : (\gamma_1, ..., \gamma_n) \in \mathbb{R} \to \sum_{i=1}^{n} \gamma_i \tag{5.2.5}$$

and the sets

$$\begin{aligned} \Omega_i &:= [0, \gamma_i^{\max}(\rho_{i,0})], \qquad i = 1, ..., n \\ \Omega_j &:= [0, \gamma_j^{\max}(\rho_{j,0})], \qquad j = n+1, ..., n+m \end{aligned} \tag{5.2.6}$$

$$\Omega := \{(\gamma_1, ..., \gamma_n) \in \gamma_1 \times \cdots \times \gamma_n | A \cdot (\gamma_1, ..., \gamma_n)^T \in \gamma_{n+1} \times \cdots \times \gamma_{n+m}\}$$

where the functions $\gamma_i^{\max}$ and $\gamma_j^{\max}$ are respectively defined in (4.2.2) and in (4.2.3).

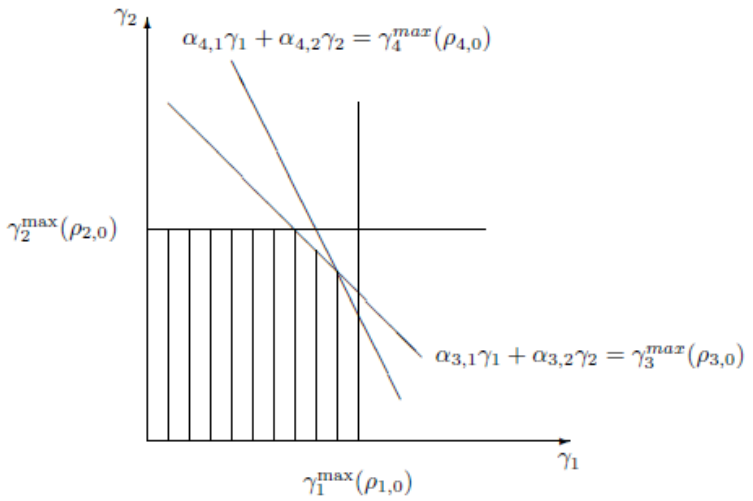

Figure 5.2.2: The set Ω for a simple junction J with 2 incoming and 2 outgoing roads.

By Proposition 4.2.2, the sets Ω_i and Ω_j contain all the possible fluxes for the solution of the Riemann problem at J. The set Ω is closed, convex and not-empty (Figure 5.2.2).

By (C), $\nabla E = \mathbf{1}$ and is not orthogonal to any nontrivial subspace contained in a supporting hyperplane of Ω, which means that there exists a unique vector $(\hat{\gamma}_1, ..., \hat{\gamma}_n) \in \Omega$ such that

$$E(\hat{\gamma}_1, ..., \hat{\gamma}_n) = \max_{(\gamma_1,...,\gamma_n)\in\Omega} E(\gamma_1, ..., \gamma_n).$$

For every $i \in \{1, ..., n\}$, we choose $\hat{\rho}_i \in [0, 1]$ such that

$$f(\hat{\rho}_i) = \gamma_i, \quad \hat{\rho}_i \in \begin{cases} \{\rho_{i,0}\} \cup (\tau(\rho_{i,0}), 1], & \text{if} \quad 0 \le \rho_{i,0} \le \sigma, \\ (\sigma, 1], & \text{if} \quad \sigma \le \rho_{i,0} \le 1. \end{cases}$$

By (A3) and (A4), a such $\hat{\rho}_i$ exists and is unique. Let

$$\hat{\gamma}_j \doteq \sum_{i=1}^{n} \alpha_{ji}\hat{\gamma}_i, \quad j = n+1, ..., n+m$$

and $\hat{\rho}_j \in [0, 1]$ be such that

$$f(\hat{\rho}_j) = \gamma_j, \quad \hat{\rho}_j \in \begin{cases} [0, \sigma], & \text{if} \quad 0 \le \rho_{j,0} \le \sigma, \\ \{\rho_{j,0}\} \cup [0, \tau(\rho_{j,0})), & \text{if} \quad \sigma \le \rho_{j,0} \le 1. \end{cases}$$

Since $(\hat{\gamma}_1, ..., \hat{\gamma}_n) \in \Omega$, $\hat{\rho}_j$ exists and is unique for every $j \in \{n+1, ..., n+m\}$. Solving the Riemann problem on each road, and the proof follows. □

The case of $n \geq 2$ Incomming Roads and $m = 1$ Outgoing Road

If there are a junction with n incoming roads and one outgoing road then the condition (C) on matrix A do not hold. If all cars can not drive through the junction, there is a yielding rule that describes the percentage of cars crossing the junction, which comes from a particular road.
We first look at the case $n = 2$. In situations like this we need to fix a right of way parameter $q \in (0, 1)$ and the rule:

(P) Assume that not all cars can enter the outgoing road and let C be the amount that can. Then qC cars come from the first incoming road and $(1-q)C$ cars from the second.

Take a junction with two incoming roads $[a_i, b_i]$, $i = 1, 2$, and one outgoing road $[a_3, b_3]$ and assume that a right of way parameter $q \in (0, 1)$ is given. Then the solution of the Riemann problem $(\rho_{1,0}, \rho_{2,0}, \rho_{3,0})$ is formed by a single wave on each road connecting the initial states to $(\hat{\rho}_1, \hat{\rho}_2, \hat{\rho}_3)$ determined in the following way.
We want to maximize the flux so we set:

$$\hat{\gamma}_3 = \min\{\gamma_1^{\max}(\rho_{1,0}) + \gamma_2^{\max}(\rho_{2,0}), \gamma_3^{\max}(\rho_{3,0})\} \tag{5.2.7}$$

where the functions $\gamma_i^{\max}$ are from (4.2.2) og (4.2.3). Consider the space (γ_1, γ_2) and the line:

$$\gamma_2 = \frac{1-q}{q}\gamma_1. \tag{5.2.8}$$

The line is the locus of points satisfying rule (P). The point of intersection between the line (5.2.8) and the line $\gamma_1 + \gamma_2 = \hat{\gamma}_3$ we denote by P. Remember that the final fluxes should belong to the region

$$\Omega = \{(\gamma_1, \gamma_2) : 0 \leq \gamma_i \leq \gamma_i^{\max}(\rho_{i,0}), 0 \leq \gamma_1 + \gamma_2 \leq \hat{\gamma}_3\}.$$

We look at the two cases:

(a) P belongs to Ω,
(b) P is outside Ω.

For the first case we set $(\hat{\gamma}_1, \hat{\gamma}_2) = P$, in the second case we set $(\hat{\gamma}_1, \hat{\gamma}_2) = Q$, where the point Q is the point of segment $Q \cap \{(\gamma_1, \gamma_2) : \gamma_1 + \gamma_2 = \hat{\gamma}_3\}$ closest to the line (5.2.8). Figure 5.2.3 shows the two cases.

For case (b) it is impossible to follow rule (P) in an exact way if we want to maximize the flux. So the point Q is the point that best follows the rule (P) in the set of points that maximize the sum of fluxes.
Once we have determined $\hat{\gamma}_1$, $\hat{\gamma}_2$ and $\hat{\gamma}_3$ we can determine ρ_i in a unique way ($i \in \{1, 2, 3\}$). From this we can deduce the following theorem:

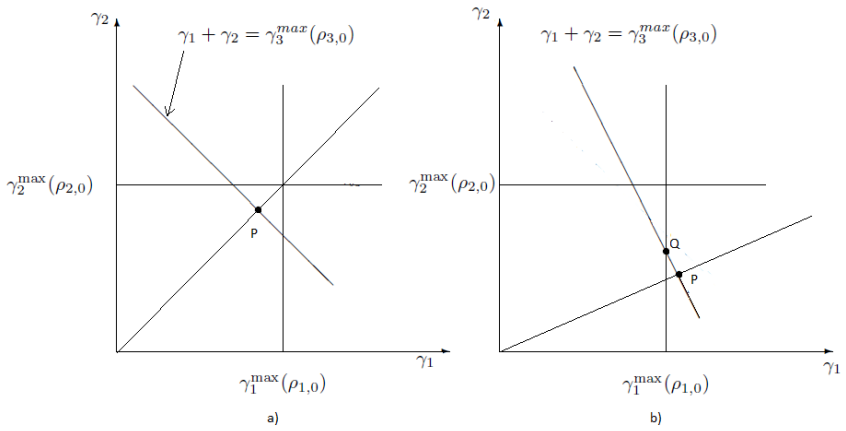

Figure 5.2.3: The case (a) and (b)

Theorem 5.2.2. *Consider a junction J with $n=2$ incoming roads and $m=1$ outgoing road, assume (A1)-(A4) and fix a right of way parameter $q\in(0,1)$. For every $\rho_{0,1},\rho_{0,2},\rho_{0,3}\in[0,1]$, there exists a unique admissible centered weak solution $\rho=(\rho_1,\rho_2,\rho_3)$ to (5.1.1) at the junction J, in the sense of Definition 5.1.4, satisfying rule (P) (possibly in a approximate way) such that*

$$\rho_{0,1}(0,\cdot)\equiv\rho_{1,0},\ \ \rho_{0,2}(0,\cdot)\equiv\rho_{2,0},\ \ \rho_{0,3}(0,\cdot)\equiv\rho_{3,0}.$$

Moreover, there exists a unique 3-tuple $(\hat\gamma_1,\hat\gamma_2,\hat\gamma_3)\in[0,1]^3$ such that

$$\hat\rho_i\in\begin{cases}\{\rho_{i,0}\}\cup(\tau(\rho_{i,0}),1], & \text{if}\quad 0\le\rho_{i,0}\le\sigma,\\ [\sigma,1], & \text{if}\quad \sigma\le\rho_{i,0}\le 1.\end{cases}\quad i=1,2, \tag{5.2.9}$$

and

$$\hat\rho_3\in\begin{cases}[0,\sigma], & \text{if}\quad 0\le\rho_{3,0}\le\sigma,\\ \{\rho_{3,0}\}\cup[0,\tau(\rho_{3,0})), & \text{if}\quad \sigma\le\rho_{3,0}\le 1.\end{cases} \tag{5.2.10}$$

and for $i\in\{1,2\}$. The solution is given by the wave $(\rho_{i,0},\hat\rho_i)$, while for the outgoing road the solution is given by the wave $(\hat\rho,\rho_{3,0})$.

Corollary 5.2.2. *Consider a junction J with $n=2$ incoming roads and $m=1$ outgoing road, assume (A1)-(A4) and fix a right of way parameter $q\in(0,1)$. Then there exists a unique Riemann solver RS, compitable with Definition 5.1.4 and rule (P). And for every $\rho_{1,0},\rho_{2,0},\rho_{3,0}\in[0,1]$, the 3-tuple $(\hat\rho_1,\hat\rho_2,\hat\rho_3)=RS(\rho_{1,0},\rho_{2,0},\rho_{3,0})$ satisfies (5.2.9) and (5.2.10).*

We describe the case of a junction J with $n>2$ incoming roads and $m=1$ outgoing road briefly. Fix $n-1$ positive parameters $q_1,...,q_{n-1}$ and consider the line r in $\mathbb{R}^n$, given by

$$\begin{cases}\gamma_n=q_1\gamma_1\\ \ \ \vdots\qquad\ \vdots\\ \gamma_n=q_{n-1}\gamma_{n-1}.\end{cases} \tag{5.2.11}$$

Here the solution to the Riemann problem with initial conditions given by $(\rho_{1,0}, ..., \rho_{n,0}, \rho_{n+1,0})$ is formed by waves connecting the initial states to $(\hat{\rho}_1, ..., \hat{\rho}_n, \hat{\rho}_{n+1})$, determined in the following way. Define

$$\hat{\gamma}_{n+1} = \min\{\gamma_1^{\max}(\rho_{1,0}) + \cdots + \gamma_n^{\max}(\rho_{n,0}), \gamma_{n+1}^{\max}(\rho_{n+1,0})\}$$

where the functions $\gamma_i^{\max}$ are defined in (5.2.2)+(5.2.3). Define the closed and convex set K in $\mathbb{R}^n$

$$\{(\gamma_1, ..., \gamma_n) : \gamma_1 + \gamma_2 + \cdots \gamma_n = \hat{\gamma}_{n+1}, 0 \leq \gamma_i \leq \hat{\gamma}_i^{\max}(\rho_{i,0}), i = 1, ..., n\}$$

Consider the unique point $(\hat{\gamma}_1, ..., \hat{\gamma}_n) \in K$ which minimizes the distance from the point $P \in r$, where P is the intersection between the line r and the hyperplane

$$\gamma_1 + \cdots + \gamma_n = \hat{\gamma}_{n+1}.$$

Finally imposing $f(\hat{\rho}_l) = \hat{\gamma}_l$ $(l = 1, ..., n, n+1)$, we obtain the trace of the solution to the Riemann problem at the junction.

6 Numerical approximations for conservation laws

We first compare the Upwind scheme and the Lax-Friedrichs scheme, which will give motivation for the Godunov method, and illustrate with some examples. All the schemes can be found and is discussed in [6].
We discretize the (x,t)-plane by the mesh (x_i, t_n) with

$$x_i = ih \quad (i \in \mathbb{Z}) \quad t_n = nk \quad (n \in \mathbb{N}_0)$$

and k,$h > 0$. For simplicity we use a uniform mesh with k and h constant, but the discussed methods can be extended to non-uniform meshes. We want a finite difference approximations u_i^n to the solution $u(x_i, t_n)$ at the discrete grid points. $i = 1, ..., N$.

6.1 Approximations for linear equations

Before we study numerical methods for *nonlinear* scalar equations we start with a very simple *linear* equation

$$u_t + au_x = 0, \quad x \in \mathbb{R}, \ t > 0, \tag{6.1.1}$$

$$u(x,0) = u_0(x), \quad x \in \mathbb{R}, \tag{6.1.2}$$

where $a > 0$. This problem has the explicit solution $u(x,t) = u_0(x - at)$ which is a weak solution (if u_0 is smooth enough).
This equation can be written by Taylor expansion as

$$\frac{u(x_i, t_{n+1}) - u(x_i, t_t)}{k} + O(k) = -a\frac{u(x_{i+1}) - u(x_{i-1})}{2h} + O(h^2),$$

which motivates the following numerical scheme:

Central scheme

$$u_i^{n+1} = u_i^n - \frac{ak}{2h}(u_{i+1}^n - u_{i-1}^n). \tag{6.1.3}$$

As we compute u_i^{n+1} from the data u_i^n, this is an *explicit* scheme. Since for time-dependent hyperbolic equations, implicit schemes are rarely used, we consider in the following only explicit schemes. The scheme can be improved by using an arithmetic average in the approximation of the time derivative. This leads to the following scheme.

Lax-Friedrichs scheme In this scheme the time derivative is approximated by

$$\frac{1}{k}\left(u(x, t+k) - \frac{1}{2}(u(x+h, t) + u(x-h, t))\right),$$

i.e

$$u_i^{n+1} = \frac{1}{2}(u_{i+1}^n + u_{i-1}^n) - \frac{ak}{2h}(u_{i+1}^n - u_{i-1}^n), \quad i = 1, ..., N-1. \tag{6.1.4}$$

The spatial derivative at x_i uses the information at x_{i+1} where the wave will go in the next time step. It would be more reasonable to use the information at x_{i-1} where the wave comes from. This is done in the following scheme.

Upwind scheme The scheme reads as follows

$$u_i^{n+1} = u_i^n - \frac{ak}{h}(u_i^n - u_{i-1}), \quad i = 1, ..., N. \tag{6.1.5}$$

This scheme gives the correct solution, no oscillations, but with artificial diffusion.
In Figure 6.1.1 we illustrate the behavior of the 3 suggested methods using discontinuous data

$$u_0(x) = \begin{cases} 1 & : \quad 0 \le x < 1/2 \\ 0 & : \quad 1/2 \le x \le 1 \end{cases} \tag{6.1.6}$$

with the parameter $a = 1$, $t = 0.25$, $k = 0.001$. We restrict the computational domain to $[0, 1]$ and the boundary data $u_0^{n+1} = u_0^n$ and $u_N^{n+1} = u_N^n$. In the traffic flow interpretation, the traffic is heavy in $[0, 1/2]$ and light in $[1/2, 1]$. So, at $x = 0$, at $x = 1$ they are exiting.

Observe that the central scheme is oscillating with damped oscillations for small k (Figure 6.1.1 (first row, left); $h = 0.01$); the Lax-Fredrichs scheme is less diffusive but not oscillatory (Figure 6.1.1(first row, right)); and the upwind scheme is less diffusive than the Lax-Friedrichs scheme (Figure 6.1.1(last row,left)). Choosing mesh size $h = k = 0.01$, both schemes produce a solution which is very close to the exact solution (Figure 6.1.1(last row,right)). All the schemes are able to compute the correct shock speed.

6.2 Approximations for nonlinear equations

The traffic flow model is nonlinear, so we want to study what happens when we discretize nonlinear equatios. We discretize the LWR-Model.

$$u_t + (u - u^2)_x = 0, \quad x \in \mathbb{R}, \ t > 0$$
$$u(x, 0) = u_0(x), \quad x \in \mathbb{R}$$

and we choose the initial values

$$u_0(x) = \begin{cases} 1/4 & : \quad x < 0 \\ 1/2 & : \quad x > 0 \end{cases} \tag{6.2.1}$$

and the boundary data $u_0^{n+1} = u_0^n$ and $u_N^{n+1} = u_N^n$. We present and compare two different numerical approximations.

Upwind Scheme

$$u_i^{n+1} = u_i^n - \frac{k}{h}(1 - 2u_{i-1}^n)(u_i^n - u_{i-1}^n) \tag{6.2.2}$$

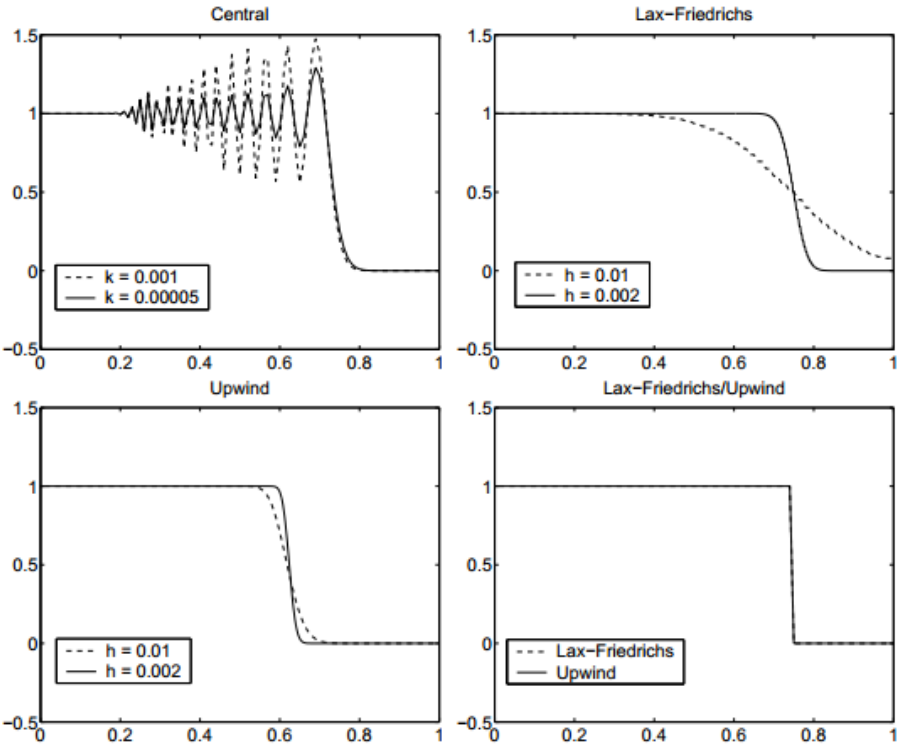

Figure 6.1.1: Various numerical schemes for (6.1.1) with $a = 1$ and discontinous data 6.1.6. First row, left : central: first row, right: Lax-Friedrich; last row, left: upwind; last row, right:Lax-Friedrichs (broke line) and upwind (solid line) for $h = 0.001$, $k = 0.001$.

Lax-Friedrichs Scheme

$$u_i^{n+1} = \frac{1}{2}(u_{i+1}^n + u_{i-1}^n) - \frac{k}{2h}((u_{i+1}^n - (u_{i+1}^n)^2) - (u_{i-1}^n - (u_{i-1}^n)^2)), \quad (6.2.3)$$

The exact solution of the problem is

$$u(x,t) = \begin{cases} 1/4 & : \quad x < \frac{1}{4}t \\ 1/2 & : \quad x > \frac{1}{4}t \end{cases}$$

so at at $t = 2$ the discontinuity should be at $x = \frac{1}{2}$. Thus the numerical solution of the Upwind scheme propagates with the wrong speed. A better behavior is given by the Lax-Friedrichs scheme. From the Figure 6.2.1 we see that the Lax-Friedrichs the scheme is non-oscillatory and the shock speed is correct.
What is the reason for the different shock speeds? The upwind scheme (6.2.1) is a discretization of the quasilinear equation

$$u_t + (1 - 2u)u_x = 0,$$

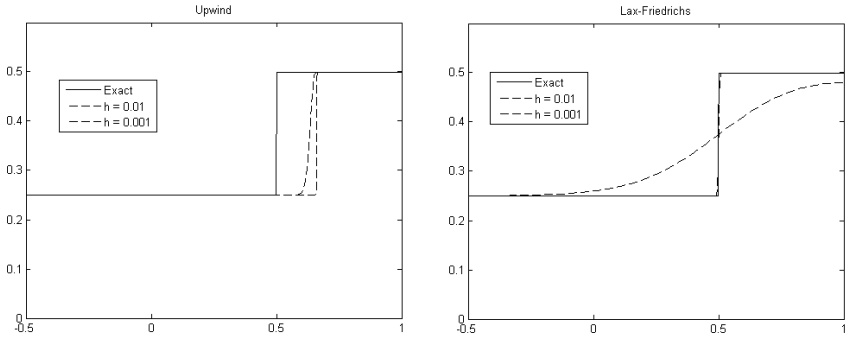

Figure 6.2.1: Exact and numerical solutions for the inviscid Burger equation using the upwind scheme(6.2.2)(left) and the Lax-Friedrichs scheme(6.2.3)(right). $k = 0.001$.

whereas the scheme (6.2.3) is an approxiation of the eqtuation in conservation form

$$u_t + (u - u^2)_x = 0.$$

For smooth solutions both equations are the same, but we know from section 2 that is may not be true for weak solutions.
In the following we consider only numerical methods in conservation form, meaning that the scheme is of the form

$$u_i^{n+1} = u_i^n - \frac{k}{h}[F(u_{i-p}^n, ..., u_{i+p}^n) - F(u_{i-1-p}^n, ..., u_{i-1+q}^n)]$$

for some function F of $p + q + 1$ arguments. We call F the *numerical flux function*. The simplest case is for $p = 0$ and $q = 1$, where

$$u_i^{n+1} = u_i^n - \frac{k}{h}[F(u_i^n, u_{i+1}^n) - F(u_{i-1}^n, u_i^n)]. \tag{6.2.4}$$

This expression can be interpreted as a cell average. We will see that the weak solution of

$$u_t + f(u)_x = 0, \quad x \in \mathbb{R},\ t > 0,$$

satisfies the integral form

$$\frac{1}{h}\int_{x_{i-1/2}}^{x_{i+1/2}} u(x, t_{n+1})dx = \frac{1}{h}\int_{x_{i-1/2}}^{x_{i+1/2}} u(x, t_n)dx - \frac{k}{h}\left[\frac{1}{k}\int_{t_n}^{t_{n+1}} f(u(x_{i+1/2}, t))dt - \frac{1}{k}\int_{t_n}^{t_{n+1}} f(u(x_{i-1/2}, t))dt\right], \tag{6.2.5}$$

where $x_{i\pm1/2} = (i \pm 1/2)h$ is the middle point between x_i and $x_{i\pm1}$. Interpreting u_i^n as an approximation of this middle point,

$$u_i^n \sim \frac{1}{h}\int_{x_{i-1/2}}^{x_{i+1/2}} u(x,t_n)dx,$$

and $F(u_i^n, u_{i+1}^n)$ as approximations of the average flux through $x_{i+1/2}$ over the time interval (t_n, t_{n+1}),

$$F(u_i^n, u_{i+1}^n) \sim \frac{1}{h}\int_{t_n}^{t_{n+1}} f(u(x_{i+1/2},t))dt,$$

we obtain the approximation (6.2.4) from (6.2.5).
The Lax-Friedrichs scheme is written in conservative form by setting:

$$F(u_i^n, u_{i+1}) = \frac{h}{2k}(u_i^n + u_{i+1}^n) + \frac{1}{2}(f(u_i^n + f(u_{i+1}^n)).$$

The conservative form of the upwind scheme is shown in the next section. A conservative scheme is called consistent if $F(u,...,u) = f(u)$.

6.3 The Godunov Method

Now we have seen some of the advantages and disadvantages of both the Lax-Friedrich scheme and upwind scheme, both for linear and nonlinear equations. We want a scheme that has Lax-Friedrichs shock speed, and that is as little dissipative as the upwind scheme. A natural generalization for the upwind scheme is

$$u_i^{n+1} = u_i^n - \frac{k}{h}[F(u_i^n, u_{i+1}^n) - F(u_{i-1}^n, u_i^n)] \tag{6.3.1}$$

$$F(v,w) = \begin{cases} f(v) & : \quad (f(v)-f(w))/(v-w) \geq 0 \\ f(w) & : \quad (f(v)-f(w))/(v-w) < 0. \end{cases}$$

For linear equations, $F(v,w) = f(v)$, and (6.3.1) reduses to the upwind scheme. However, there is a problem with the above approximation. Take $f(u) = u(1-u)$ and choose the initial values:

$$u_i^0 = \begin{cases} 0 & : \quad i \leq 0 \\ 1 & : \quad i > 0. \end{cases}$$

Since $F(u_i^0, u_{i+1}^0)$ and $F(u_{i-1}^0, u_i^0)$ are either equal to $f(0)$ or $f(1)$ and since $f(0) = f(1)$, we obtain that $u_i^1 = u_i^0$ for all i and hence $u_i^n = u_i^0$ for all i. which is not correct.
This leads to the Godunov Scheme, a conservative and consistent generalization of the upwind scheme. Let f be a concave C^2 function. The idea of this method is to approximate the solution $u(x,t_n)$ of the conservation law (scalar or system)

$$u_t + f(u)_x = 0, \qquad x \in \mathbb{R},\ t > 0,$$

by i piecewise constant fuction $\tilde{u}^n(x,t_n)$ by solving the Riemann problem in the interval $t \in [t_n, t_{n+1}]$. After obtaining this solution, we define the approximate solution u_i^{n+1} at time t_{n+1} by averaging at the time t_{n+1}:

$$u_i^{n+1} = \frac{1}{h}\int_{x_{i-1/2}}^{x_{i+1/2}} \tilde{u}^n(x,t_{n+1})dx, \tag{6.3.2}$$

where $x_{i\pm 1/2} = (i \pm 1/2)h$. These values are then used to define the new piecewise constant data $\tilde{u}^{n+1}(x,t_{n+1})$ by:

$$\tilde{u}^{n+1}(x,t_{n+1}) = u_i^{n+1} \quad if \quad x_{i-1/2} \le x < x_{i+1/2}$$

and the process repeats.
We can considerably simplify this algorithm since the above integral can be computed explicitly. Since $\tilde{u}^n$ is assumed to be the exact weak solution, it satisfies a weak solution proposed in [1], but we present it without the technical proof:
Let u be a classical solution of (2.2.1). Integrating (2.2.1) over $(x_1,x_2)\times(s,t)$ for any $(x_1,x_2) \in \mathbb{R}$ and $s,t > 0$ gives

$$\int_{x_1}^{x_2} u(x,t)dx - \int_{x_1}^{x_2} u(x,s)dx = -\int_s^t f(u(u(x_2,\tau)))d\tau + \int_s^t f(u(x_1,\tau))d\tau.$$

Inserting $\tilde{u}^n$ and divided by h we get:

$$\begin{aligned}\tfrac{1}{h}\textstyle\int_{x_{i-1/2}}^{x_{i+1/2}} \tilde{u}^n(x,t_{n+1})dx \quad &= \tfrac{1}{h}\textstyle\int_{x_{i-1/2}}^{x_{i+1/2}} \tilde{u}^n(x,t_n)dx \\ &\quad -\tfrac{k}{h}\left[\tfrac{1}{k}\textstyle\int_{t_n}^{t_{n+1}} f(\tilde{u}^n(x_{i+1/2},t))dt - \tfrac{1}{k}\textstyle\int_{t_n}^{t_{n+1}} f(\tilde{u}^n(x_{i-1/2},t))dt\right]\end{aligned}$$

From (6.3.2) it follows that

$$u_i^{n+1} = u_i^n - \frac{k}{h}[F(u_i^n, u_{i+1}^n) - F(u_{i-1}^n, u_i^n)],$$

where the numerical flux function F is given by

$$F(u_i^n, u_{i+1}^n) = \frac{1}{k}\int_{t_n}^{t_{n+1}} f(\tilde{u}^n(x_{i+1/2},t))dt.$$

So the Godunov scheme is conservative. The function $\tilde{u}$ is constant on the line $x = x_{i+1/2}, t_n \le t \le t_{n+1}$, see Figure 6.3.1. We denote this value by $u^*(u_i^n, u_{i+1}^n)$.

The flux then reduces to

$$F(u_i^n, u_{i+1}^n) = f(u^*(u_i^n, u_{i+1}^n))$$

and the Godunov scheme becomes

$$u_i^{n+1} = u_i^n - \frac{k}{h}[f(u^*(u_i^n, u_{i+1}^n)) - f(u^*(u_{i-1}^n, u_i^n))].$$

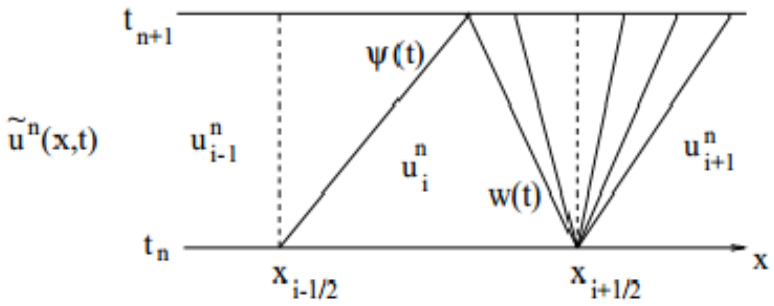

Figure 6.3.1: Illustration for the Godunov scheme. There is a shock through $x_{i-1/2}$ and a rarefraction wave starting at $x_{i+1/2}$.

The scheme is consistent since $F(u_i^n, u_i^n) = f(u_i^n)$ and f is assumed to be smooth.
For large $t - t_n$, the solution may not remain constant at $x_{i+1/2}$ because of the effects of waves arising from neighbouring Riemann problems. How large can we choose $k = t_{n+1} - t_n$? Assume the situation of Figure 6.3.1, i.e let $\psi(t)$ be the shock line through $x_{i-1/2}$ and let $\omega(t)$ be the left end of the rarefraction wave starting at $x_{i+1/2}$. The time t_{n+1} is determined by the requirement $\psi(t_{n+1}) \leq \omega(t_{t+1})$. Since

$$\psi(t) = x_{i-1/2} + s(t - t_n) \quad with \quad s = \frac{f(u_{i+1}^n) - f(u_i^n)}{u_{i+1}^n - u_i^n}$$

and

$$\omega(t) = x_{i+1/2} + f'(u_i^n)(t - t_n),$$

this means that

$$h = x_{i+1/2} - x_{i-1/2} \geq (s - f'(u_i^n))(t_{n+1} - t_n) = (s - f'(u_i^n))k. \tag{6.3.3}$$

As f is assumed to be concave, s lies between $f'(u_i^n)$ and $f'(u_{i+1}^n)$, i.e

$$|s| \leq \max\{|f'(u_i^n)|, |f'(u_{i+1}^n)|\}.$$

we get

$$s - f'(u_{i+1}^n) \leq |s| + |f'(u_i^n)| \leq 2\sup_{i,n}|f'(u_i^n)| \leq \frac{h}{k}$$

so (6.3.3) is satisfied. This condition ensures that the shock and the rarefraction wave do not interact in the mesh cell $[x_{i-1/2}, x_{i+1/2}] \times [t_n, t_{n+1})$. We obtain the same condition if there is a rarefraction wave at $x_{i-1/2}$ and a shock at $x_{i+1/2}$ or if there are two shocks at $x_{i\pm1/2}$ since the wave speed are bounded by $\sup |f'(u_i^n)|$.

We can allow the waves to interact during the time step, provided the interaction is entirely contained within a mesh cell. This leads to the condition $2 \sup |f'(u_i^n)| \leq 2h/k$ or

$$\nu = \sup_{i,n} \left| f'(u_i^n) \frac{k}{h} \right| \leq 1,$$

This condition can be interpreted as a generalization of the CFL condition for linear conservation laws.
Our next question is how can we determine u^* from u_i^n, u_{i+1}^n? We need to solve the Riemann problem. For scalar conservation laws, we can determine u^* easily from the signs of $f'(u_i^n)$ and $f'(u_{i+1}^n)$. We have to consider the four cases:

(a) $f'(u_i^n) \geq 0$ and $f'(u_{i+1}^n) \geq 0$: In this case there is a rarefraction wave starting at $x_{i+1/2}$ and from Figure 6.3.2 (a) we see that $u^* = u_i^n$.

(b) $f'(u_i^n) < 0$ and $f'(u_{i+1}^n) < 0$: Again there is a rarefraction wave starting at $x_{i+1/2}$ but now $u^* = u_{i+1}^n$.(Figure 6.3.2 (b).)

(c) $f'(u_i^n) \geq 0$ and $f'(u_{i+1}^n) < 0$: There is a shock through $x_{i+1/2}$ and (Figure 6.3.2 (c))

$$u^* = \begin{cases} u_i^n & : \quad s \geq 0 \\ u_{i+1}^n & : \quad s < 0. \end{cases}$$

(d) $f'(u_i^n) < 0$ and $f'(u_{i+1}^n) \geq 0$: There is a rarefracton wave starting at $x_{i+1/2}$ and u^* is the unique solution of $f'(u^*) = 0$ (since f is concave). (Figure 6.3.2 (d)).
The resulting flux function can be written in the simplified form:

$$F(u_i^n, u_{i+1}^n) = \begin{cases} \min\limits_{u_i^n \leq u \leq u_{i+1}^n} f(u) & : \quad u_i^n \leq u_{i+1}^n \\ \min\limits_{u_{i+1}^n \leq u \leq u_i^n} f(u) & : \quad u_i^n > u_{i+1}^n. \end{cases}$$

This expression holds for general conservation laws, even non-concave ones, and gives the correct Godunov flux corresponding to the weak solution satisfying the entropy condition (2.3.1) of Oleinik.

Example 6.3.1. *For $f(u) = au (a > 0)$ the flux function becomes*

$$F(u_i^n, u_{i+1}^n) = au_i^n,$$

hence the Godunov scheme is equal to the upwind scheme for linear equations.

$$u_i^{n+1} = u_i^n - \frac{ak}{h}(u_i^n - u_{i-1}^n)$$

In this way, we can say that the Godunov method is a generalization of the upwind scheme to nonlinear equations.

More information about the Godunov Scheme can be found in [1], [7] and [9].

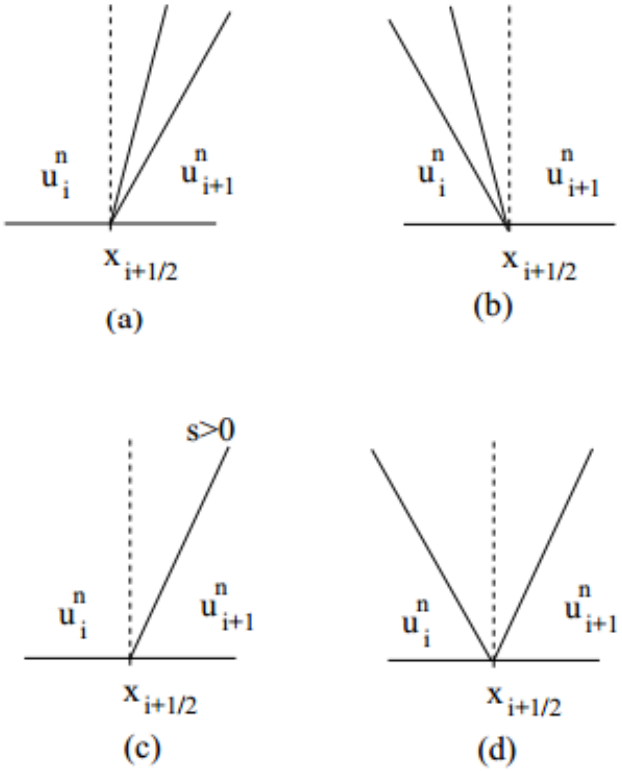

Figure 6.3.2: How to compute u^*.

6.4 Boundary Conditions and Conditions at Junctions

To set up a Godunov scheme we need to impose some boundary conditions and some conditions at the junctions.

Boundary conditions

Suppose we assign a condition at the incoming boundary $x = 0$:

$$u(0,t) = \rho_1(t), \quad t > 0$$

and study the equation only for $x > 0$. We are considering the initial-boundary value problem

$$\begin{aligned} &u_t + f(u)_x = 0, \\ &u(x,0) = u_0(x), \qquad x \geq 0, \\ &u(0,t) = u_b(t), \qquad t \geq 0, \end{aligned}$$

In general the boundary data cannot be assumed, so it is not easy to find a function u that satisfies the boundary value $u(0,t) = u_b(t)$ in a classical sense.

We need to look for a condition that is effective only in the inflow part of the boundary. Following [2], the way to assign the boundary condition is

$$\max_{k\in I(u(0,t),\rho_1(t))}\{sign(u(0,t)-\rho_1(t))[F(u(0,t))-F(k)]\}=0 \tag{6.4.1}$$

We inserti a ghost cell and define

$$u_0^{n+1}=u_0^n-\frac{k}{h}[F(u_0^n,u_1^n)-F(v_1^n,u_0^n)]$$

where

$$v_1^n(t)=\frac{1}{k}\int_{t_n}^{t_{n+1}}\rho_1 dt \tag{6.4.2}$$

takes the place of u_{-1}^n. An outgoing boundary can be treated analogously. Let $x<L=x_N$. Then the discretization reads:

$$u_N^{n+1}=u_0^N-\frac{k}{h}[F(u_N^n,v_2^n)-F(u_{N-1}^n,u_N^n)], \tag{6.4.3}$$

where

$$v_2(t)=\frac{1}{k}\int_{t_n}^{t_{n+1}}\rho_2 dt$$

takes the place of u_{N+1}^n, the ghost cell value. (See [2],[9])

Conditions at a junction

For roads connected to a junction at the right endpoint we set

$$u_N^{n+1}=u_0^N-\frac{k}{h}[\hat{\gamma}_i-F(u_{N-1}^n,u_N^n)]$$

while for roads connected to a junction at the left endpoint we have

$$u_0^{n+1}=u_0^n-\frac{k}{h}[F(u_0^n,u_1^n)-\hat{\gamma}_j]$$

where $\hat{\gamma}_i$,$\hat{\gamma}_j$ are the maximized fluxes.
For the Godunov's scheme there is no need to invert the flux f to put it into the scheme, as the Godunov's flux coincides with the Riemann's flux. In this case it suffices to insert the computed maximized fluxes directly in the scheme.

Motion of the cars

We now have a matrix with different densities for all times $t_n=nk$ and all positions $x_i=ih$ where $n=1,..,T$, and $i=1,...,N$. We want to know how a car starting at some (x_i,t_n) moves as time passes. We do it in the following way: assuming that at (x_i,t_n) the cars has the speed $v(\rho(x_i,t_n))$ and it will have that speed until it reaches x_{i+1}. We then check how long time it takes, and gives the car the new speed at that point. Say it takes $t=3k$, then the new speed is $v(\rho(x_{i+1},t_{n+3}))$, and it keeps this speed until x_{i+2} and the process continues.

7 Numerical Exampels using the Godunov Method

In this chapter we start by solving the Riemann problem on a straight road. Then we look at a narrowing, i.e. a road that connects with a road with a smaller flux function. The final test is a system with multiple roads and junctions.

7.1 Riemann problem

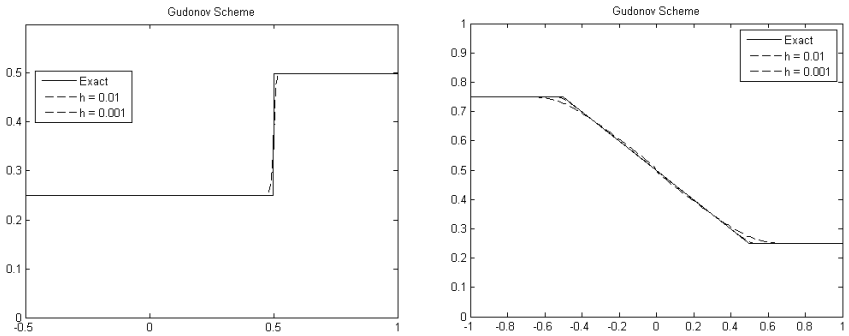

Figure 7.1.1: Godunov Scheme for a shock solution at $T = 2$ (left) and Rarefraction at $T = 1$ (right). $k = 0.001$.

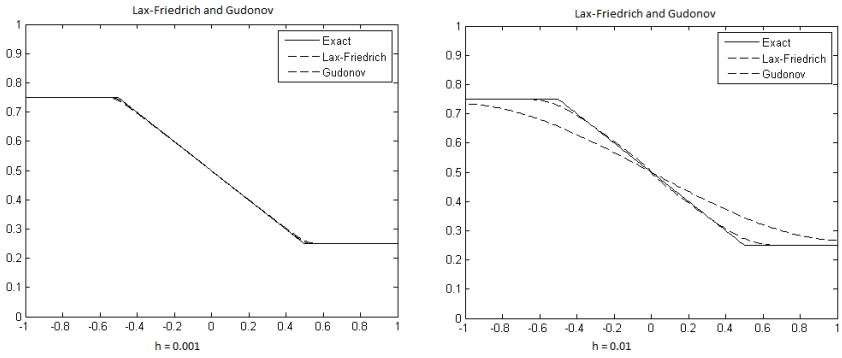

Figure 7.1.2: Godunov and Lax-friedrich for a Rarefraction solution $h = 0.001$(left) and h = 0.01(right). $k = 0.001$ and $T = 1$.

In this section we will look at the Riemann problem, a simple road with two densities. Consider the case where a road of length 2 is paramterized on the interval $[-1, 1]$. On the left side of zero the road has density ρ_1 and on the right side of zero it has the density ρ_2. We use the LWR-model

$$\rho_t + f(\rho)_x = 0,$$

where $f(\rho) = \rho(1-\rho)$.

Rarefraction wave $\rho_1 > \rho_2$. Consider the initial values

$$\rho(x,0) = \begin{cases} \frac{3}{4}, & x < 0 \\ \frac{1}{4}, & x > 0 \end{cases}$$

Then the solutions for $t > 0$ becomes

$$\rho(x,t) = \begin{cases} \frac{3}{4}, & x < -\frac{1}{2}t \\ \frac{1}{2}\left(1 - \frac{x}{t}\right), & -\frac{1}{2}t \leq x \leq \frac{1}{2}t \\ \frac{1}{4}, & x > \frac{1}{2}t \end{cases}$$

Shock Solution $\rho_1 < \rho_2$. Consider the initial values

$$\rho(x,0) = \begin{cases} \frac{1}{4}, & x < 0 \\ \frac{1}{2}, & x > 0 \end{cases}$$

Then the shock speed is $s = 1/4$ and the solutions for $t > 0$ becomes

$$\rho(x,0) = \begin{cases} \frac{1}{4}, & x < \frac{1}{4}t \\ \frac{1}{2}, & x > \frac{1}{4}t. \end{cases}$$

We can se from the Figure 7.1.2 that the Godunov scheme is more precise than the Lax-Friedrich for smaller values of h when we deal with the rarefraction solutions, and it also gives better approximations for $h = 0.001$ even though its hard to see from the picture. If we compare Figure 7.1.1 with Figure 6.2.1 it is more precise than both Lax-Friedrich and the upwind scheme for shock solutions.

7.2 Narrowing

The simplest example with a junction is two roads, one incoming and one outgoing, where the outgoing has a smaller flux than the incoming one. We can look at this as a road being narrowed. We set the flux in the first road to be equal to

$$f_1(\rho) = \rho(1-\rho), \quad \rho \in [0,1] \tag{7.2.1}$$

and the flux in the second road to be

$$f_2 = \rho(1-2\rho), \quad \rho \in [0, \frac{1}{2}] \tag{7.2.2}$$

The maximum fluxes is unique:

$$f_1(\sigma_1) = \max_{[0,1]} f_1(\rho) = \frac{1}{4}, \ with\ \sigma_1 = \frac{1}{2} \tag{7.2.3}$$

$$f_2(\sigma_2) = \max_{[0,1/2]} f_2(\rho) = \frac{1}{8}, \ with\ \sigma_2 = \frac{1}{4} \tag{7.2.4}$$

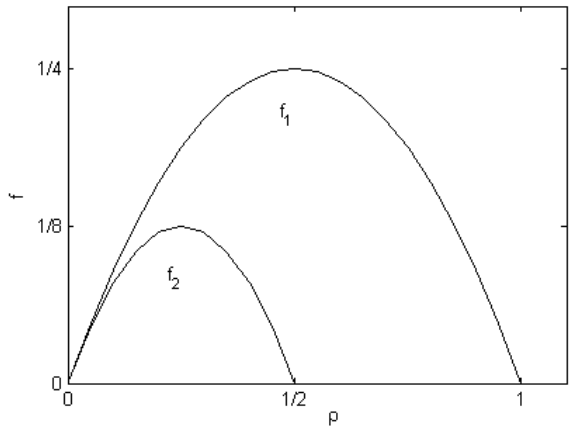

Figure 7.2.1: The flux functions $f_1(\rho)$ and $f_2(\rho)$.

We call the point of separation between the two roads S. We indicate by ρ_l the density on the left side of S (the widest part of the street) and by ρ_r the density of the right side of S.
The maximal fluxes f_1 and f_2 are computed:

$$f_1^{\max}(\rho) = \begin{cases} f_1(\rho_l) & \text{if} \quad \rho_l \leq \sigma_1, \\ f_1(\sigma_1) & \text{if} \quad \rho_l \geq \sigma_1, \end{cases}$$

$$f_2^{\max}(\rho) = \begin{cases} f_2(\sigma_2) & \text{if} \quad \rho_r \leq \sigma_2, \\ f_2(\rho_r) & \text{if} \quad \rho_r \geq \sigma_2 \end{cases}$$

and the maximal flux at the intersection point between the two intervals is obtained by taking the minimum

$$\gamma = \min\{f_1^{\max}, f_2^{\max}\}. \tag{7.2.5}$$

The creation of queues occurs when the density of the first road verifies

$$\rho(1-\rho) = \frac{1}{8} \iff \bar{\rho} = \frac{1 - \sqrt{\frac{1}{2}}}{2} \approx 0.15. \tag{7.2.6}$$

If the car density entering the largest road, say $\rho_{1,b}$, is such that $\rho_{1,b} < \bar{\rho}$ there is no formation of shocks propagating backwards. If $\rho_{1,b} > \bar{\rho}$ there will be a traffic jam.

We now present an example. We consider a road of length 2 parameterized by the interval $[0, 2]$ and set s to be at $x = 1$. Consider the following initial values:

$$\rho_l = 0, \quad \rho_r = 0 \quad \rho_{1,b} = \frac{1}{2}. \tag{7.2.7}$$

This can interpreted a red light turning green in at $x = 0$, and that the road goes from two lanes to one in $x = 1$. Since $\rho 1, b > \bar{\rho}$ we get a jam. Se Figure 7.2.2.

We estimate the L^1 error in the following way:

$$e^r = \sum_{j=1}^{N} |u_j^M(h) - u_{2j}^M(\frac{h}{2})|, \qquad r = 1, ..., R.$$

$u_m^M(h)$ is the numerical solution with space step discretization equal to h, in x_m at the final time $t_M = T$. R denotes the number of roads. Then

$$TOT_{error} = \sum_{r=1}^{R} e^r.$$

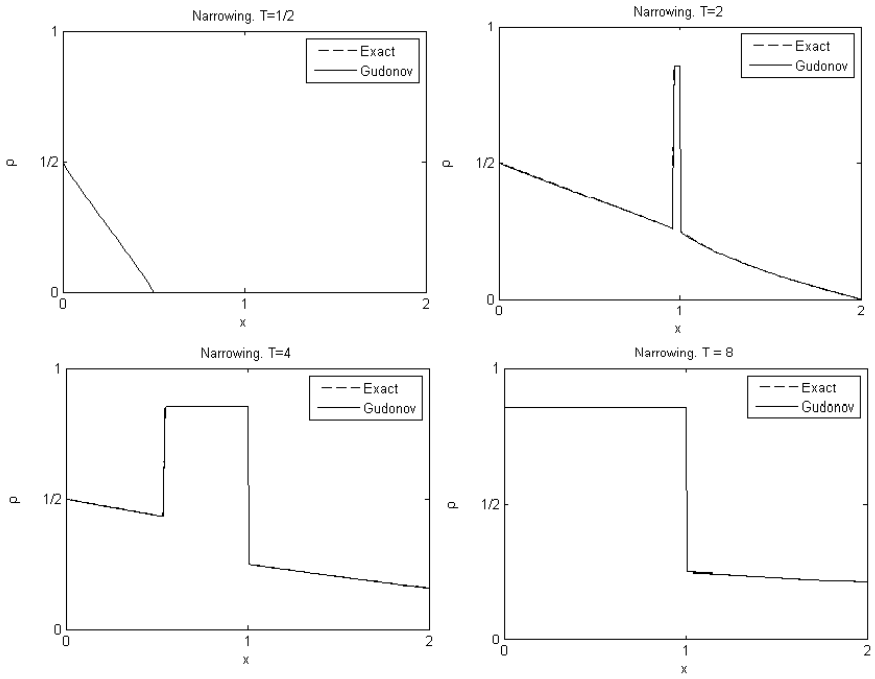

Figure 7.2.2: Godunov Scheme for the narrowing problem at $T = 0.5$ (top left). $T = 2$ (top right). $T = 4$ (bottom left). $T = 8$ (bottom right). $k = h = \frac{1}{160}$. The anlytical solution is plotted with a strippeled line.

	L^1 Error	L^1 Error	L^1 Error	L^1 Error
h↓ T→	1/2	2	4	8
0.1	$2.0390e-002$	$3.1164e-002$	$2.6270e-002$	$1.2223e-003$
0.05	$2.1203e-002$	$1.4799e-002$	$9.2414e-003$	$7.8577e-004$
0.025	$1.8090e-002$	$1.0529e-002$	$7.1166e-003$	$4.8417e-004$
0.0125	$1.3398e-002$	$3.3735e-003$	$2.7080e-003$	$2.8410e-004$
0.00625	$9.0833e-003$	$2.4058e-003$	$1.5887e-003$	$1.6137e-004$
0.003125	$5.8004e-003$	$1.7093e-003$	$1.1270e-003$	$8.8344e-005$

Table 1: L^1-error approximated numerically for the Narrowing problem, with different times and values of h. $k = h$.

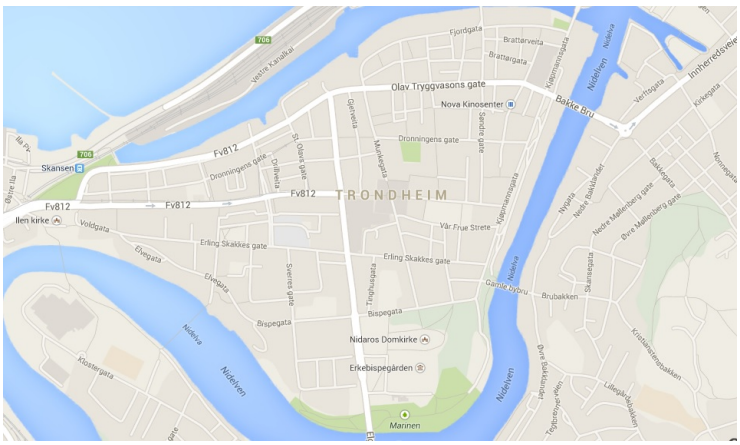

Figure 7.3.1: The center of Trondheim.

7.3 Trondheim

Now we present a simplified model of downtown Trondheim (Figure 7.3.1).

We use the roads from Figure 7.3.2. The red roads are the main roads, while the pink roads are smaller side roads. We will model this in three different ways, and give two examples for each model. They will be discussed in some detail, but watching the videos[11] will give the best impression of how the density evolves in the different models. All videos last until $T = 35$. For each test there will be a tables and graphs showing how long time it takes to drive different routes. Notice that the routes can be of different lengths. The red roads will be numbered $1, .., n$ for some n, depending on the model, and $s1 - s4$ are the numbers for the pink roads. We choose the functions f_1 and f_2 from the previous example (7.2.1)-(7.2.2), where f_1 is used on the red roads $1 - n$ and f_2 is used on the pink roads $s1 - s4$. To simplify all roads are of length 1, and is discretisized on the intervall $[0, 1]$ with $h = k = 0.025$. Let the initial car

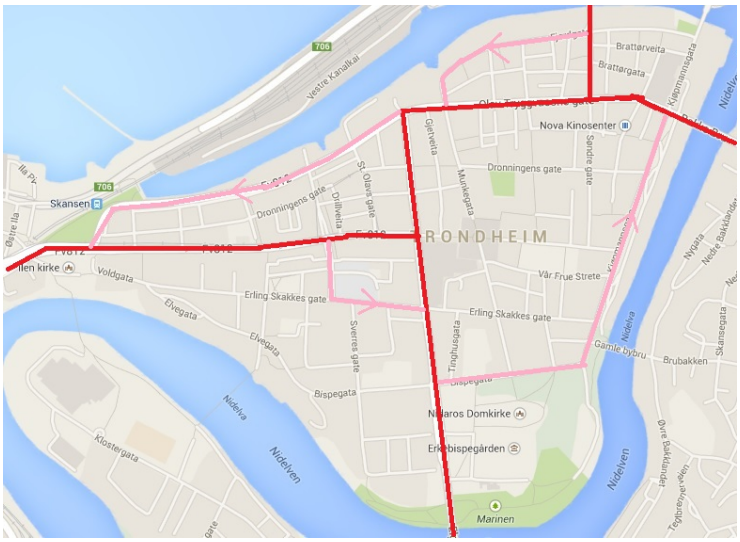

Figure 7.3.2: The roads that are to be used.

density the roads be

$$\rho_0 = (\rho_1(0,x), ..., \rho_n(0,x), \rho_{s1}(0,x), ..., \rho_{s4}(0,x)).$$

Model A

Model A is based on Figure 7.3.3. This model consists of ten junctions, where five of them have one incoming and two outgoing roads, the other five have two incoming and one outgoing road. So 17 roads in total, 13 main roads and 4 side roads. In this model we look at the traffic going south. When a big roads divides into one small and one big, we set the distribution matrix to be

$$A = \begin{pmatrix} 0.7 \\ 0.3 \end{pmatrix}.$$

Hence 70 percent of the drivers choose the big road, while 30 percent choose the small road. If it divides into two roads with the same size the traffic is divided evenly. When two roads of different size meet and become one, we set the right of way parameter to be $q = 0.70$, in favour of the bigger roads since they are more trafficated. If they are the same size we set $q = 0.5$.

Test A1. Let us assume that the roads are initially empty and take the following initial and boundary data:

$$\rho_0 = (0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)$$

$$\rho_{3,b}(t,0) = 0.5 \qquad \rho_{7,b}(t,0) = 0.5$$

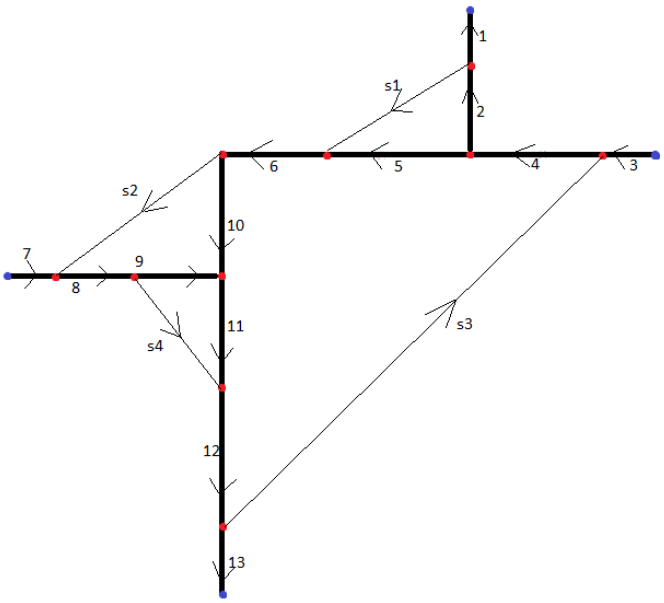

Figure 7.3.3: Model A of Trondheim. Southbound traffic.

Comments about the video: Rarefraction waves start at the incoming roads 3 and 7, and spread through the system. Just before $T = 6$ the wave has gone through roads s3 and s4, causing the incoming fluxes on road 4 and 8 to reach their maximum. This results in shocks moving backwards on road 3 and 7. Right after $T = 8$ road 12 reaches its incoming max, and we see a shock on road 11. More and more roads are being filled up with cars, and shocks move backwards. At $T = 22$ the shock from road 9 goes straight through road 8 causing a second shock on road 7. Furthermore, a slow shock goes through 10, reaching road 6. More cars from road 6 choose the side road s2. The shock on road 6 will slowly fill up roads 4 and 3, so that road 2 reaches $\rho = 0.25$ and road 1 reaches $\rho = 0.175$.

Test A2. Assume that the main roads are heavy trafficated, and that the side roads have some traffic on them. And we assume that there are little incoming traffic to Trondheim. We choose the following initial and boundary data:

$$\rho_0 = (0.4, 0.5, 0.5, 0.7, 0.60, 0.7, 0.5, 0.5, 0.5, 0.8, 0.7, 0.7, 0.4, 0.1, 0.15, 0.2, 0.1)$$

$$\rho_{3,b}(t, 0) = 0.15 \qquad \rho_{7,b}(t, 0) = 0.15$$

Comments about the video: At first, shock- and rarefraction waves start at almost all endpoints of the roads. At around $T = 2$ the shocks have met. Most main roads give shocks backwards, filling up the roads one by one. Roads 1,2,3 and 4 are having shocks backwards. At $T = 20$ road 4 starts feeding road 5 with less cars and a shock is moving forward, emptying the road. As the

density increases on road s2, a shock goes through road 6 and hits the one on road 5. The new shock will empty the remaining roads one by one, but it moves so slowly that it is 1/4 into road 6 at $T = 150$.

Motion of cars: We check the driving duration through each route starting at different times(T) for both tests. The routes we use are

$$\begin{aligned} \textit{Route } 1 &: 3 \to 4 \to 2 \to s1 \to 6 \to s2 \to 8 \to s4 \to 12 \to 13 \\ \textit{Route } 2 &: 3 \to 4 \to 5 \to 6 \to 10 \to 11 \to 12 \to 13 \\ \textit{Route } 3 &: 3 \to 4 \to 5 \to 6 \to s2 \to 8 \to s4 \to 12 \to 13 \end{aligned}$$

A1	T=0	T=5	T=10	A2	T=0	T=5	T = 10
Route 1	15.33	18.08	20.18	Route 1	32.18	32.19	33.18
Route 2	11.00	16.36	20.22	Route 2	36.31	35.01	34.37
Route 3	13.16	16.58	18.29	Route 3	34.56	35.06	35.21

Table 2: Route through model A. Initial values from test A1 and A2. T =0, 5 and 10.

Model B

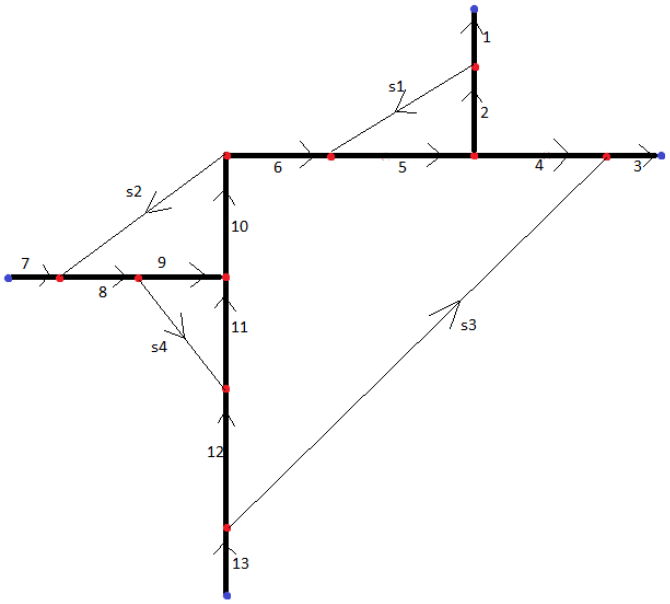

Figure 7.3.4: Model B of Trondheim. Northbound traffic.

Model B is based on Figure 7.3.4. It is the same model as Model A, except that roads $3, 4, 5, 6, 10, 11$ and 12 have changed direction. We will use the same rules

at junctions as Model A for the distribution of traffic.

Test B1. Let us assume that the roads are initially empty and take the following initial and boundary data:

$$\rho_0 = (0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)$$

$$\rho_{13,b}(t,0) = 0.5 \qquad \rho_{7,b}(t,0) = 0.5$$

Comments about the video: The cars spread out from road 7 and road 13. The first cars enter the outgoing roads 3 and 1 at around $T = 7$. The only backward shock we get in this test is in road 7 just before $T = 6$. As the time increases roads 10,11 and 12 reach $\rho = 0.5$, while other main roads have a slightly lower ρ. So driving through traffic in this model is pretty smooth, as long as you avoid road 7 until $T = 7$.

Test B2. Let us assume that the roads are initially empty and take the following initial and boundary data:

$$\rho_0 = (0.45, 0.5, 0.45, 0.5, 0.5, 0.7, 0.55, 0.3, 0.5, 0.5, 1, 0.5, .45, 0.25, 0, 0, 0.25)$$

$$\rho_{13,b}(t,0) = 0.15 \qquad \rho_{7,b}(t,0) = 0.25$$

Comments about the video: Just like model A2 this model starts with many different shocks and rarefractions. From $T = 1.8$ to $T = 2.5$ notice that road s1 has a shock going straight up. Just as in model B1, a queue is formed at road 7. Then most roads get less and less dense. Road 8 is being filled up until it reaches $\rho = 0.5$. Road 10 has $\rho = 0.5$ during the whole simulation. Hence roads 7, 8 and 10 are the most trafficated.

Motion of cars: We check the driving duration through each route starting at different times(T) for both tests. For model B we use the following routes:

$$\begin{aligned} \textit{Route } 1 &: 7 \to 8 \to s4 \to 11 \to 10 \to 6 \to 5 \to 2 \to 1 \\ \textit{Route } 2 &: 7 \to 8 \to 9 \to 10 \to 6 \to 2 \to 1 \end{aligned}$$

B1	T=0	T=5	T=10	B2	T=0	T=5	T = 10
Route 1	12.68	16.34	18.56	Route 1	22.66	20.44	18.06
Route 2	8.53	12.93	14.09	Route 2	15.44	15.05	14.55

Table 3: Route through model B. Initial values from test B1 and B2. T=0, 5 and 10.

Model C

Model C is based on Figure 7.3.5. In this model the red roads go both ways. This model has ten junctions: two junctions with 3 incoming and 3 outgoing

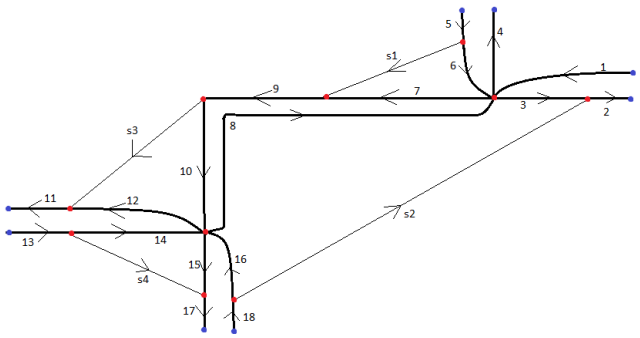

Figure 7.3.5: Model C of Trondheim. Red roads have two directions.

roads, four junctions with 2 incoming and 1 outgoing, and four junctions with 1 incoming and 2 outgoing. For the eight junctions with 3 roads we use the same rules as in Model A and Model B. But for the junctions with 6 roads we choose the following distribution matrix:

$$A = \begin{pmatrix} \alpha_{3,1} & \alpha_{3,6} & \alpha_{3,8} \\ \alpha_{7,1} & \alpha_{7,6} & \alpha_{7,8} \\ \alpha_{4,1} & \alpha_{4,6} & \alpha_{4,8} \end{pmatrix} = \begin{pmatrix} \alpha_{8,10} & \alpha_{8,14} & \alpha_{8,16} \\ \alpha_{15,10} & \alpha_{15,14} & \alpha_{15,16} \\ \alpha_{12,10} & \alpha_{12,14} & \alpha_{12,16} \end{pmatrix} = \begin{pmatrix} 0.1 & 0.5 & 0.4 \\ 0.5 & 0.4 & 0.1 \\ 0.4 & 0.1 & 0.5 \end{pmatrix}.$$

This distribution matrix, which respects condition (C), makes sure that only a small amount of cars choose the same way they came from, and that most traffic is directed through town. It also prevents a lot of traffic getting stuck in the loop created by road 8 and 9.

Test C1. Let us assume that the roads is initially empty and take the following initial and boundary data:

$$\rho_0 = (0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0)$$

$$\rho_{1,b}(t,0) = 0.5 \qquad \rho_{5,b}(t,0) = 0.5 \qquad \rho_{13,b}(t,0) = 0.5 \qquad \rho_{18,b}(t,0) = 0.5$$

Comments about the video: There is not much to comment on this video. We get the first shock on road 7 just after $T = 6$. This shock continues through road 1, as this has the least right of way. As most roads end up at $\rho = 0.5$ or just under, roads 1,3 and 7 will have some heavier traffic. Roads 3 and 7 ends up at $\rho = 0.7738$.

Test C2. Let us assume that the main roads are crowded and take the initial values where there are few cars entering the system:

$$\rho_0 = (0.35, 0.35, 0.2, 0.4, 0.35, 0.4, 0.7, 0.7, 0.6, 0.5, 0.25, 0.5, 0.25, 0.5, 0.4, 0.7, 0.45, 0.35, 0.1, 0.3, 0.4, 0)$$

$$\rho_{1,b}(t,0) = 0 \qquad \rho_{5,b}(t,0) = 0.45 \qquad \rho_{13,b}(t,0) = 0.15 \qquad \rho_{18,b}(t,0) = 0.25$$

Comments about the video: There is a lot going on in the beginning of model C2. The amount of cars flowing into the model is low, and there are a lot of cars in the system. We notice that cars a leaving roads 1 and 13, and that 2 and 11 has more incoming that outgoing. Other roads start with a shock on both endpoints. When the shocks meet, a new shock is created, where the "bigger" shock wins. At $T = 13$, only roads 2 and 11 have max incoming flux, and there is a small queue at road 5. Roads 3,9 and 12 have forwards shocks. The rest of the roads, except s3, are stable at a low densitis. Road s3 is almost full, but this changes when the shock from road 9 reaches it. After this we only see forward shocks for some of the roads, while other stay stable. The whole system stabilizes at $T = 27$.

Motion of cars: We check the driving duration through each route starting at different times(T) for both tests. Routes for model C:

$$\begin{aligned} \textit{Route } 1 &: 5 \to s1 \to 9 \to s3 \to 11 \\ \textit{Route } 2 &: 5 \to 6 \to 7 \to 9 \to 10 \to 12 \to 11 \\ \textit{Route } 3 &: 18 \to 16 \to 8 \to 7 \to 9 \to s3 \to 11 \\ \textit{Route } 4 &: 18 \to 16 \to 8 \to 7 \to 9 \to 10 \to 12 \to 11 \end{aligned}$$

C1	T=0	T=5	T=10	T=25	C2	T=0	T=5	T = 10	T = 25
Route 1	6.50	9.35	9.58	9.75	Route 1	14.01	14.13	13.08	8.50
Route 2	10.00	14.17	15.23	16.75	Route 2	18.24	15.38	13.40	11.75
Route 3	10.25	14.25	15.38	16.50	Route 3	19.99	16.68	14.84	11.00
Route 4	11.75	16.39	17.38	18.50	Route 3	19.62	16.15	14.15	12.50

Table 4: Routes through model C. With initial values from test C1 and C2. T=0,5,10 and 25.

Figures

Plotting time vs. positions of the different routes of the models of Trondheim. Initial values from all the tests (pp. 60-62).

8 Comments and future work

On the web page [10], I have put my matlab codes. It is possible to change all the fluxes and right of way parameters if you want look at how traffic evolves in other situations. The code used for creating the videoes are also in the folder. I have tried my best to provide enough comments in the codes so that it will be understandable.

We know from [2] that there are better ways to model a traffic network than the Godunov method. A three velocity kinetic approximation of second order($3VK_2$) have been presented and tested on a model of a traffic circle. The $3VK_2$-model has a smaller error than the Godunov scheme, but it is harder to implement. In my code I have managed to conserve the flux in all of the different types of junctions. Thus it can be a useful tool in analysing the behavior of traffic in larger networks.

As for the testing of the different routes, I would in hindsight have inserted an extra road or two so that the comparison between two routes would be more interesting. I.e. more routes starting from the same junction, ending at the same, and see for what situations it would be better to choose one over an other. For my tests the same route is always the fastest for all starting times.

One natural step for future work can be to modify the program to take in roads of different lengths, and with different velocity functions. It would not be to hard, but I decided to use roads of equal length as this simplifies the code and makes it easier for the reader to understand. The same goes for different velocity functions. For bigger cities it would seem more realistic to have time based traffic lights at junctions, instead of a right of way parameters. This can be done by setting the boundary condition at junctions with red lights to zero, while using the max flux function for roads with green lights. Traffic lights have been discussed in [1], [2] and [5].

9 Pseudocode for the Godunov Scheme

$$u_i^0 = \frac{1}{h}\int_{x_{i-1/2}}^{x_{i+1/2}} u_0(x)dx.$$

For all i,n:

$$\begin{array}{llllll}
f'(u_i^n) \geq 0 & \text{and} & f'(u_{i+1}^n) \geq 0 & \text{then} & u_i^* = u_i^n; \\
f'(u_i^n) < 0 & \text{and} & f'(u_{i+1}^n) < 0 & \text{then} & u_i^* = u_{i+1}^n, \\
f'(u_i^n) \geq 0 & \text{and} & f'(u_{i+1}^n) < 0 & \text{then} & u_i^* = u_i^n \ if\ (s \geq 0)\ or\ u_i^* = u_{i+1}^n\ (if\ s < 0) \\
f'(u_i^n) < 0 & \text{and} & f'(u_{i+1}^n) \geq 0 & \text{then} & u_i^*\ is\ the\ unique\ solution\ of\ f'(u_i^*) = 0.
\end{array}$$

$$u_i^{n+1} = u_i^n - \frac{k}{h}[f(u_i^*) - f(u_{i-1}^*)].$$

References

[1] A. Jüngel *Modeling and Numerical Approximation of Traffic Flow Problems*. Universität Mainz, Preliminary notes, 2002.

[2] B. Piccoli, M.Garavello *Traffic Flow on Networks*, AIMS, 2006.

[3] A. Aw and M. Rascle *Resurrection of second order models of traffic flow*. SIAM J. Appl. Math, 2000.

[4] J.N. McDonald, N.A.Weiss *A Course in Real Analysis, Second Edition*, Academic Press, 2012.

[5] H.E. Krogstad *Modeling Based on Conservation Principles*, Department of Mathematical Sciences, NTNU, 2011.

[6] B. Owren *Numerisk løsning av partielle differensialligninger med endelig differansemetode*, Department of Mathematical Sciences, NTNU, 2007.

[7] R. LeVeque *Numerical Methods for Conservation Laws*. Birkhuser, Basel, 1990.

[8] H.Holden, N. H. Risebro *Fron Tracking for Hyperbolic Conservation Laws*. Applied Mathematical Sciences, 150, Springer, Berlin 2002.

[9] S. K. Godunov *A finite difference method for the numerical computation of discontinuous solutions of the equations of fluid dynamics*, Mat. Sb,. 1959.

[10] B. Bergersen *http://folk.ntnu.no/bjornard/master/*
2014,
Files: 'ModelA.m','ModelB.m', 'ModelC.m', 'LaxFriedrish.m' ,'Narrowing.m', 'Riemannproblem.m', 'FilmModelAB.m', 'FilmModelC'.

[11] B. Bergersen *https://www.youtube.com/user/ModelingTraffic*
2014

10 Appendix A: Matlab codes

Because of the size of the codes only the code for model C is included in the paper. As mentioned the other codes can be found in [11].

Model C

```
%---From the figure Model C of Trondheim.
%---Everything above line 38 are initial values and can be changed.
clear;
h = 0.025;              % Step i x-direction
T = 35;                 % Time steps
q =.70;                 % Right of way parameter for roads junctions with 2
      incomming and 1 outgoing roads.
alpha1 =.70;            % Procentage of cars from big roads to big roads in
      junctions with 1 incomming and 2 outgoing.
alpha2 =.30;            % Procentage of cars from big roads to small roads in
      junctions with 1 incomming and 2 outging.
%-------- Initial values of the 22 roads
u1  = .35;
u2  = .35;
u3  = .2;
u4  = .4;
u5  = .35;
u6  = .4;
u7  = .7;
u8  = .7;
u9  = .6;
u10 = .5;
u11 = .25;
u12 = .5;
u13 = .25;
u14 = .5;
u15 = .4;
u16 = .7;
u17 = .45;
u18 = .35;
%--"sideroads"--
us1 = .1;
us2 = .3;
us3 = .4;
us4 =  0;
%---- Flux on the 4 incomming roads.
u1b  =   0;
u5b  = .45;
u13b = .15;
u18b = .25;
%-------------------------------------------------------------------------
A = [.1 .5 .4;...   % Distribution matrix for the junctions with 3 incomming
      and 3 outgoing roads.
     .5 .4 .1;...
     .4 .1 .5];
k = h;
x = 0:h:1;          % Points of the roads
Nx = length(x);  % number of steps in x-direction
t = floor(T/k);  % number of steps in t-direction
%Starting flux on the roads (F1 is road 1 etc.)
F1(1,1:Nx) = u1;
F2(1,1:Nx) = u2;
F3(1,1:Nx) = u3;
F4(1,1:Nx) = u4;
F5(1,1:Nx) = u5;
F6(1,1:Nx) = u6;
F7(1,1:Nx) = u7;
F8(1,1:Nx) = u8;
F9(1,1:Nx) = u9;
F10(1,1:Nx) = u10;
F11(1,1:Nx) = u11;
F12(1,1:Nx) = u12;
F13(1,1:Nx) = u13;
F14(1,1:Nx) = u14;
F15(1,1:Nx) = u15;
F16(1,1:Nx) = u16;
F17(1,1:Nx) = u17;
F18(1,1:Nx) = u18;
Fs1(1,1:Nx) = us1;
Fs2(1,1:Nx) = us2;
Fs3(1,1:Nx) = us3;
Fs4(1,1:Nx) = us4;

for i = 1:t
```

```
    v = [0;0;0];
    v2 = [0;0;0];
    %-----find u* = U(i,1) for the 4 incoming roads-----
    a = 1-2*u1b;
    b = 1-2*F1(i,1);
    s = 1-u1b-F1(i,1);
    if a >= 0 && b >= 0
            U01 = u1b;
        elseif a <0 && b <0
            U1 = F1(i,1);
        elseif a >= 0 && b < 0 && s >= 0
            U01 = u1b;
        elseif a >= 0 && b <0 && s < 0
            U01= F1(i,1);
        elseif a <0 && b >= 0
            U01 = .5;
    end
    a = 1-2*u5b;
    b = 1-2*F5(i,1);
    s = 1-u5b-F5(i,1);
    if a >= 0 && b >= 0
            U05 = u5b;
        elseif a <0 && b <0
            U05 = F5(i,1);
        elseif a >= 0 && b < 0 && s >= 0
            U05 = u5b;
        elseif a >= 0 && b <0 && s < 0
            U05= F5(i,1);
        elseif a <0 && b >= 0
            U05 = .5;
    end

    a = 1-2*u13b;
    b = 1-2*F13(i,1);
    s = 1-u13b-F13(i,1);
    if a >= 0 && b >= 0
            U013 = u13b;
        elseif a <0 && b <0
            U013 = F13(i,1);
        elseif a >= 0 && b < 0 && s >= 0
            U013 = u13b;
        elseif a >= 0 && b <0 && s < 0
            U013= F13(i,1);
        elseif a <0 && b >= 0
            U013 = .5;
    end
    a = 1-2*u18b;
    b = 1-2*F18(i,1);
    s = 1-u18b-F18(i,1);
    if a >= 0 && b >= 0
            U018 = u18b;
        elseif a <0 && b <0
            U018 = F18(i,1);
        elseif a >= 0 && b < 0 && s >= 0
            U018 = u18b;
        elseif a >= 0 && b <0 && s < 0
            U018= F18(i,1);
        elseif a <0 && b >= 0
            U018 = .5;
    end

    %-- Find u* F1:
   for n = 1:Nx-1
        a = 1-2*F1(i,n);              % f'(u(i,n))
        b = 1-2*F1(i,n+1);            % f'(u(i,n+1))
        s = 1- F1(i,n+1) - F1(i,n); % Rankine-Hugoniot condition

        if a >= 0 && b >= 0
            U1(i,n) = F1(i,n);
        elseif a <0 && b <0
            U1(i,n) = F1(i,n+1);
        elseif a >= 0 && b < 0 && s >= 0
            U1(i,n) = F1(i,n);
        elseif a >= 0 && b <0 && s < 0
            U1(i,n)= F1(i,n+1);
        elseif a <0 && b >= 0
            U1(i,n) = .5;
        end
    end
   %-- Find u* F2:
    for n = 1:Nx-1
        a = 1-2*F2(i,n);
        b = 1-2*F2(i,n+1);
        s = 1- F2(i,n+1) - F2(i,n);

        if a >= 0 && b >= 0
            U2(i,n) = F2(i,n);
        elseif a <0 && b <0
            U2(i,n) = F2(i,n+1);
        elseif a >= 0 && b < 0 && s >= 0
            U2(i,n) = F2(i,n);
```

```
        elseif a >= 0 && b <0 && s < 0
            U2(i,n)= F2(i,n+1);
        elseif a <0 && b >= 0
            U2(i,n) = .5;
        end
    end
    %-- Find u* F3:
    for n = 1:Nx-1
        a = 1-2*F3(i,n);
        b = 1-2*F3(i,n+1);
        s = 1- F3(i,n+1) - F3(i,n);

        if a >= 0 && b >= 0
            U3(i,n) = F3(i,n);
        elseif a <0 && b <0
            U3(i,n) = F3(i,n+1);
        elseif a >= 0 && b < 0 && s >= 0
            U3(i,n) = F3(i,n);
        elseif a >= 0 && b <0 && s < 0
            U3(i,n)= F3(i,n+1);
        elseif a <0 && b >= 0
            U3(i,n) = .5;
        end
    end
    %-- Find u* F4:
    for n = 1:Nx-1
        a = 1-2*F4(i,n);
        b = 1-2*F4(i,n+1);
        s = 1-F4(i,n+1) - F4(i,n);

        if a >= 0 && b >= 0
            U4(i,n) = F4(i,n);
        elseif a <0 && b <0
            U4(i,n) = F4(i,n+1);
        elseif a >= 0 && b < 0 && s >= 0
            U4(i,n) = F4(i,n);
        elseif a >= 0 && b <0 && s < 0
            U4(i,n)= F4(i,n+1);
        elseif a <0 && b >= 0
            U4(i,n) = .5;
        end
    end
    %-- Find u* F5:
    for n = 1:Nx-1
        a = 1-2*F5(i,n);
        b = 1-2*F5(i,n+1);
        s = 1- F5(i,n+1) - F5(i,n);

        if a >= 0 && b >= 0
            U5(i,n) = F5(i,n);
        elseif a <0 && b <0
            U5(i,n) = F5(i,n+1);
        elseif a >= 0 && b < 0 && s >= 0
            U5(i,n) = F5(i,n);
        elseif a >= 0 && b <0 && s < 0
            U5(i,n)= F5(i,n+1);
        elseif a <0 && b >= 0
            U5(i,n) = .5;
        end
    end
    %-- Find u* F6:
    for n = 1:Nx-1
        a = 1-2*F6(i,n);
        b = 1-2*F6(i,n+1);
        s = 1- F6(i,n+1) - F6(i,n);

        if a >= 0 && b >= 0
            U6(i,n) = F6(i,n);
        elseif a <0 && b <0
            U6(i,n) = F6(i,n+1);
        elseif a >= 0 && b < 0 && s >= 0
            U6(i,n) = F6(i,n);
        elseif a >= 0 && b <0 && s < 0
            U6(i,n)= F6(i,n+1);
        elseif a <0 && b >= 0
            U6(i,n) = .5;
        end
    end
    %-- Find u* F7:
    for n = 1:Nx-1
        a = 1-2*F7(i,n);
        b = 1-2*F7(i,n+1);
        s = 1- F7(i,n+1) - F7(i,n);

        if a >= 0 && b >= 0
            U7(i,n) = F7(i,n);
        elseif a <0 && b <0
            U7(i,n) = F7(i,n+1);
        elseif a >= 0 && b < 0 && s >= 0
            U7(i,n) = F7(i,n);
        elseif a >= 0 && b <0 && s < 0
```

```
253             U7(i,n)= F7(i,n+1);
254         elseif a <0 && b >= 0
255             U7(i,n) = .5;
256         end
257     end
258     %-- Find u* F8:
259     for n = 1:Nx-1
260         a = 1-2*F8(i,n);
261         b = 1-2*F8(i,n+1);
262         s = 1- F8(i,n+1) - F8(i,n);
263 
264         if a >= 0 && b >= 0
265             U8(i,n) = F8(i,n);
266         elseif a <0 && b <0
267             U8(i,n) = F8(i,n+1);
268         elseif a >= 0 && b < 0 && s >= 0
269             U8(i,n) = F8(i,n);
270         elseif a >= 0 && b <0 && s < 0
271             U8(i,n)= F8(i,n+1);
272         elseif a <0 && b >= 0
273             U8(i,n) = .5;
274         end
275     end
276     %-- Find u* F9:
277     for n = 1:Nx-1
278         a = 1-2*F9(i,n);
279         b = 1-2*F9(i,n+1);
280         s = 1- F9(i,n+1) - F9(i,n);
281 
282         if a >= 0 && b >= 0
283             U9(i,n) = F9(i,n);
284         elseif a <0 && b <0
285             U9(i,n) = F9(i,n+1);
286         elseif a >= 0 && b < 0 && s >= 0
287             U9(i,n) = F9(i,n);
288         elseif a >= 0 && b <0 && s < 0
289             U9(i,n)= F9(i,n+1);
290         elseif a <0 && b >= 0
291             U9(i,n) = .5;
292         end
293     end
294     %-- Find u* F10:
295     for n = 1:Nx-1
296         a = 1-2*F10(i,n);
297         b = 1-2*F10(i,n+1);
298         s = 1- F10(i,n+1) - F10(i,n);
299 
300         if a >= 0 && b >= 0
301             U10(i,n) = F10(i,n);
302         elseif a <0 && b <0
303             U10(i,n) = F10(i,n+1);
304         elseif a >= 0 && b < 0 && s >= 0
305             U10(i,n) = F10(i,n);
306         elseif a >= 0 && b <0 && s < 0
307             U10(i,n)= F10(i,n+1);
308         elseif a <0 && b >= 0
309             U10(i,n) = .5;
310         end
311     end
312     %-- Find u* F11:
313     for n = 1:Nx-1
314         a = 1-2*F11(i,n);
315         b = 1-2*F11(i,n+1);
316         s = 1- F11(i,n+1) - F11(i,n);
317 
318         if a >= 0 && b >= 0
319             U11(i,n) = F11(i,n);
320         elseif a <0 && b <0
321             U11(i,n) = F11(i,n+1);
322         elseif a >= 0 && b < 0 && s >= 0
323             U11(i,n) = F11(i,n);
324         elseif a >= 0 && b <0 && s < 0
325             U11(i,n)= F11(i,n+1);
326         elseif a <0 && b >= 0
327             U11(i,n) = .5;
328         end
329     end
330     %-- Find u* F12:
331     for n = 1:Nx-1
332         a = 1-2*F12(i,n);
333         b = 1-2*F12(i,n+1);
334         s = 1- F12(i,n+1) - F12(i,n);
335 
336         if a >= 0 && b >= 0
337             U12(i,n) = F12(i,n);
338         elseif a <0 && b <0
339             U12(i,n) = F12(i,n+1);
340         elseif a >= 0 && b < 0 && s >= 0
341             U12(i,n) = F12(i,n);
342         elseif a >= 0 && b <0 && s < 0
343             U12(i,n)= F12(i,n+1);
```

```
344          elseif a <0 && b >= 0
345              U12(i,n) = .5;
346          end
347      end
348      %-- Find u* F13:
349      for n = 1:Nx-1
350          a = 1-2*F13(i,n);
351          b = 1-2*F13(i,n+1);
352          s = 1- F13(i,n+1) - F13(i,n);
353
354          if a >= 0 && b >= 0
355              U13(i,n) = F13(i,n);
356          elseif a <0 && b <0
357              U13(i,n) = F13(i,n+1);
358          elseif a >= 0 && b < 0 && s >= 0
359              U13(i,n) = F13(i,n);
360          elseif a >= 0 && b <0 && s < 0
361              U13(i,n)= F13(i,n+1);
362          elseif a <0 && b >= 0
363              U13(i,n) = .5;
364          end
365      end
366      %-- Find u* F14:
367      for n = 1:Nx-1
368          a = 1-2*F14(i,n);
369          b = 1-2*F14(i,n+1);
370          s = 1- F14(i,n+1) - F14(i,n);
371
372          if a >= 0 && b >= 0
373              U14(i,n) = F14(i,n);
374          elseif a <0 && b <0
375              U14(i,n) = F14(i,n+1);
376          elseif a >= 0 && b < 0 && s >= 0
377              U14(i,n) = F14(i,n);
378          elseif a >= 0 && b <0 && s < 0
379              U14(i,n)= F14(i,n+1);
380          elseif a <0 && b >= 0
381              U14(i,n) = .5;
382          end
383      end
384      %-- Find u* F15:
385      for n = 1:Nx-1
386          a = 1-2*F15(i,n);
387          b = 1-2*F15(i,n+1);
388          s = 1- F15(i,n+1) - F15(i,n);
389
390          if a >= 0 && b >= 0
391              U15(i,n) = F15(i,n);
392          elseif a <0 && b <0
393              U15(i,n) = F15(i,n+1);
394          elseif a >= 0 && b < 0 && s >= 0
395              U15(i,n) = F15(i,n);
396          elseif a >= 0 && b <0 && s < 0
397              U15(i,n)= F15(i,n+1);
398          elseif a <0 && b >= 0
399              U15(i,n) = .5;
400          end
401      end
402      %-- Find u* F16:
403      for n = 1:Nx-1
404          a = 1-2*F16(i,n);
405          b = 1-2*F16(i,n+1);
406          s = 1- F16(i,n+1) - F16(i,n);
407
408          if a >= 0 && b >= 0
409              U16(i,n) = F16(i,n);
410          elseif a <0 && b <0
411              U16(i,n) = F16(i,n+1);
412          elseif a >= 0 && b < 0 && s >= 0
413              U16(i,n) = F16(i,n);
414          elseif a >= 0 && b <0 && s < 0
415              U16(i,n)= F16(i,n+1);
416          elseif a <0 && b >= 0
417              U16(i,n) = .5;
418          end
419      end
420      %-- Find u* F17:
421      for n = 1:Nx-1
422          a = 1-2*F17(i,n);
423          b = 1-2*F17(i,n+1);
424          s = 1- F17(i,n+1) - F17(i,n);
425
426          if a >= 0 && b >= 0
427              U17(i,n) = F17(i,n);
428          elseif a <0 && b <0
429              U17(i,n) = F17(i,n+1);
430          elseif a >= 0 && b < 0 && s >= 0
431              U17(i,n) = F17(i,n);
432          elseif a >= 0 && b <0 && s < 0
433              U17(i,n)= F17(i,n+1);
434          elseif a <0 && b >= 0
```

```
435             U17(i,n) = .5;
436         end
437     end
438     %-- Find u* F18:
439     for n = 1:Nx-1
440         a = 1-2*F18(i,n);
441         b = 1-2*F18(i,n+1);
442         s = 1- F18(i,n+1) - F18(i,n);
443 
444         if a >= 0 && b >= 0
445             U18(i,n) = F18(i,n);
446         elseif a <0 && b <0
447             U18(i,n) = F18(i,n+1);
448         elseif a >= 0 && b < 0 && s >= 0
449             U18(i,n) = F18(i,n);
450         elseif a >= 0 && b <0 && s < 0
451             U18(i,n)= F18(i,n+1);
452         elseif a <0 && b >= 0
453             U18(i,n) = .5;
454         end
455     end
456     %-- Find u* Fs1:
457     for n = 1:Nx-1
458         a = 1-4*Fs1(i,n);
459         b = 1-4*Fs1(i,n+1);
460         s = 1- 2*Fs1(i,n+1) - 2*Fs1(i,n);
461 
462         if a >= 0 && b >= 0
463             Us1(i,n) = Fs1(i,n);
464         elseif a <0 && b <0
465             Us1(i,n) = Fs1(i,n+1);
466         elseif a >= 0 && b < 0 && s >= 0
467             Us1(i,n) = Fs1(i,n);
468         elseif a >= 0 && b <0 && s < 0
469             Us1(i,n)= Fs1(i,n+1);
470         elseif a <0 && b >= 0
471             Us1(i,n) = 1/4;
472         end
473     end
474     %-- Find u* Fs2:
475     for n = 1:Nx-1
476         a = 1-4*Fs2(i,n);
477         b = 1-4*Fs2(i,n+1);
478         s = 1- 2*Fs2(i,n+1) - 2*Fs2(i,n);
479 
480         if a >= 0 && b >= 0
481             Us2(i,n) = Fs2(i,n);
482         elseif a <0 && b <0
483             Us2(i,n) = Fs2(i,n+1);
484         elseif a >= 0 && b < 0 && s >= 0
485             Us2(i,n) = Fs2(i,n);
486         elseif a >= 0 && b <0 && s < 0
487             Us2(i,n)= Fs2(i,n+1);
488         elseif a <0 && b >= 0
489             Us2(i,n) = 1/4;
490         end
491     end
492        %-- Find u* Fs3:
493     for n = 1:Nx-1
494         a = 1-4*Fs3(i,n);
495         b = 1-4*Fs3(i,n+1);
496         s = 1- 2*Fs3(i,n+1) - 2*Fs3(i,n);
497 
498         if a >= 0 && b >= 0
499             Us3(i,n) = Fs3(i,n);
500         elseif a <0 && b <0
501             Us3(i,n) = Fs3(i,n+1);
502         elseif a >= 0 && b < 0 && s >= 0
503             Us3(i,n) = Fs3(i,n);
504         elseif a >= 0 && b <0 && s < 0
505             Us3(i,n)= Fs3(i,n+1);
506         elseif a <0 && b >= 0
507             Us3(i,n) = 1/4;
508         end
509     end
510        %-- Find u* Fs4:
511     for n = 1:Nx-1
512         a = 1-4*Fs4(i,n);
513         b = 1-4*Fs4(i,n+1);
514         s = 1- 2*Fs4(i,n+1) - 2*Fs4(i,n);
515 
516         if a >= 0 && b >= 0
517             Us4(i,n) = Fs4(i,n);
518         elseif a <0 && b <0
519             Us4(i,n) = Fs4(i,n+1);
520         elseif a >= 0 && b < 0 && s >= 0
521             Us4(i,n) = Fs4(i,n);
522         elseif a >= 0 && b <0 && s < 0
523             Us4(i,n)= Fs4(i,n+1);
524         elseif a <0 && b >= 0
525             Us4(i,n) = 1/4;
```

```
526          end
527      end
528         %Find the flux for the next timestep for 2:Nx-1
529  for n = 2:Nx-1
530    F1(i+1,n) = F1(i,n)  -(k/h)*((U1(i,n)-U1(i,n)^2)-(U1(i,n-1)-U1(i,n-1)^2));
531    F2(i+1,n) = F2(i,n)  -(k/h)*((U2(i,n)-U2(i,n)^2)-(U2(i,n-1)-U2(i,n-1)^2));
532    F3(i+1,n) = F3(i,n)  -(k/h)*((U3(i,n)-U3(i,n)^2)-(U3(i,n-1)-U3(i,n-1)^2));
533    F4(i+1,n) = F4(i,n)  -(k/h)*((U4(i,n)-U4(i,n)^2)-(U4(i,n-1)-U4(i,n-1)^2));
534    F5(i+1,n) = F5(i,n)  -(k/h)*((U5(i,n)-U5(i,n)^2)-(U5(i,n-1)-U5(i,n-1)^2));
535    F6(i+1,n) = F6(i,n)  -(k/h)*((U6(i,n)-U6(i,n)^2)-(U6(i,n-1)-U6(i,n-1)^2));
536    F7(i+1,n) = F7(i,n)  -(k/h)*((U7(i,n)-U7(i,n)^2)-(U7(i,n-1)-U7(i,n-1)^2));
537    F8(i+1,n) = F8(i,n)  -(k/h)*((U8(i,n)-U8(i,n)^2)-(U8(i,n-1)-U8(i,n-1)^2));
538    F9(i+1,n) = F9(i,n)  -(k/h)*((U9(i,n)-U9(i,n)^2)-(U9(i,n-1)-U9(i,n-1)^2));
539    F10(i+1,n)= F10(i,n)-(k/h)*((U10(i,n)-U10(i,n)^2)-(U10(i,n-1)-U10(i,n-1)^2));
540    F11(i+1,n)= F11(i,n)-(k/h)*((U11(i,n)-U11(i,n)^2)-(U11(i,n-1)-U11(i,n-1)^2));
541    F12(i+1,n)= F12(i,n)-(k/h)*((U12(i,n)-U12(i,n)^2)-(U12(i,n-1)-U12(i,n-1)^2));
542    F13(i+1,n)= F13(i,n)-(k/h)*((U13(i,n)-U13(i,n)^2)-(U13(i,n-1)-U13(i,n-1)^2));
543    F14(i+1,n)= F14(i,n)-(k/h)*((U14(i,n)-U14(i,n)^2)-(U14(i,n-1)-U14(i,n-1)^2));
544    F15(i+1,n)= F15(i,n)-(k/h)*((U15(i,n)-U15(i,n)^2)-(U15(i,n-1)-U15(i,n-1)^2));
545    F16(i+1,n)= F16(i,n)-(k/h)*((U16(i,n)-U16(i,n)^2)-(U16(i,n-1)-U16(i,n-1)^2));
546    F17(i+1,n)= F17(i,n)-(k/h)*((U17(i,n)-U17(i,n)^2)-(U17(i,n-1)-U17(i,n-1)^2));
547    F18(i+1,n)= F18(i,n)-(k/h)*((U18(i,n)-U18(i,n)^2)-(U18(i,n-1)-U18(i,n-1)^2));
548    Fs1(i+1,n)= Fs1(i,n)-(k/h)*((Us1(i,n)-2*Us1(i,n)^2)...
549                           -(Us1(i,n-1)-2*Us1(i,n-1)^2));
550    Fs2(i+1,n)= Fs2(i,n)-(k/h)*((Us2(i,n)-2*Us2(i,n)^2)...
551                           -(Us2(i,n-1)-2*Us2(i,n-1)^2));
552    Fs3(i+1,n)= Fs3(i,n)-(k/h)*((Us3(i,n)-2*Us3(i,n)^2)...
553                           -(Us3(i,n-1)-2*Us3(i,n-1)^2));
554    Fs4(i+1,n)= Fs4(i,n)-(k/h)*((Us4(i,n)-2*Us4(i,n)^2)...
555                           -(Us4(i,n-1)-2*Us4(i,n-1)^2));
556  end
557    bout(1:22) = [F1(i,Nx) F2(i,Nx) F3(i,Nx) F4(i,Nx) F5(i,Nx) F6(i,Nx) F7(i,Nx)
          F8(i,Nx) F9(i,Nx) F10(i,Nx) F11(i,Nx) F12(i,Nx) F13(i,Nx) F14(i,Nx) F15(
          i,Nx) F16(i,Nx) F17(i,Nx) F18(i,Nx) Fs1(i,Nx) Fs2(i,Nx) Fs3(i,Nx) Fs4(i,
          Nx)];
558     %--Find the maxfluxes out. (4.2.3)
559     for l = 1:18
560        if bout(l) <= .5
561          b(l) = bout(l) -bout(l)^2;
562     else
563          b(l) = 1/4;
564        end
565     end
566     %--Find the maxfluxes out sideroads. (4.2.3)
567     for l = 19:22
568        if bout(l) <= 1/4
569          b(l) = bout(l) -2*bout(l)^2;
570     else
571          b(l) = 1/8;
572        end
573     end
574     bin(1:22) = [F1(i,1) F2(i,1) F3(i,1) F4(i,1) F5(i,1) F6(i,1) F7(i,1) F8(i,1)
            F9(i,1) F10(i,1) F11(i,1) F12(i,1) F13(i,1) F14(i,1) F15(i,1) F16(i,1)
            F17(i,1) F18(i,1) Fs1(i,1) Fs2(i,1) Fs3(i,1) Fs4(i,1)];
575     %--Find the maxfluxes in (4.2.2)
576     for L = 1:18
577     if bin(L) <= .5
578          a(L) = 1/4;
579     else
580          a(L) = bin(L)-bin(L)^2;
581     end
582     end
583     %--Find the maxfluxes in sideroads. (4.2.2)
584     for L = 19:22
585     if bin(L) <= 1/4
586          a(L) = 1/8;
587     else
588          a(L) = bin(L)-2*bin(L)^2;
589     end
590     end
591     %--Find the fluxes in and out for each junction.--
592     %Junctions with 2 insomming, 1 outgoing (4 junctions)
593     %Junction(3,s2 -> 2)
594     In2 = min(b(3)+ b(20),a(2));
595     Out3 = b(3);
596     Outs2 = b(20);
597     if b(3) + b(20) > a(2) % Amount of cars exceeds C
598     Out3 = q*In2;
599     Outs2 =(1-q)*In2;
600     if Out3 > b(3)          % if p is outside omega
601     Out3 = b(3);
602     Outs2 = In2 - b(3);
603     elseif Outs2 > b(20)  % if p is outside omega (other side)
604     Outs2 = b(20);
605     Out3 = In2 -b(20);
606     end
607     end
608     %Junction(7,s1 -> 9) 2 out, 1 in
609     In9 = min(b(7)+ b(19),a(9));
610     Out7 = b(7);
611     Outs1= b(19);
```

```
612     if b(7) + b(19) > a(19) %(Amount of cars exceeds C)
613     Out7 = q*In9;
614     Outs1 =(1-q)*In9;
615     if Out7 > b(7)   % if p lies outside omega.
616     Out7 = b(7);
617     Outs1 = In9 - b(7);
618     elseif Outs1 > b(19)  % if p lies outside omega. (other side)
619     Outs1 = b(19);
620     Out7 = In9 -b(19);
621     end
622     end
623     %Junction (12,s3 -> 11)
624     In11 = min(b(12)+ b(21),a(11));
625     Out12 = b(12);
626     Outs3 = b(21);
627     if b(12)+b(21) > a(11) %Amount of cars exceeds C.
628     Out12 = q*In11;
629     Outs3 =(1-q)*In11;
630     if Out12 > b(12)  % if p lies outside omega.
631     Out12 = b(12);
632     Outs3 = In11 - b(12);
633     elseif Outs3 > b(21) % if p lies outside omega (other side).
634     Outs3 = b(21);
635     Out12 = In11 -b(21);
636     end
637     end
638     %Junction (15,s4 -> 17)
639     In17 = min(b(15)+ b(22),a(17));
640     Out15 = b(15);
641     Outs4 = b(22);
642     if b(15) + b(22) > a(17)%(Amount of cars exceeds C)
643     Out15 = q*In17;
644     Outs4 =(1-q)*In17;
645     if Out15 > b(15)        % if p lies outside gamma.
646     Out15 = b(15);
647     Outs4 = In17 - b(15);
648     elseif Outs4 > b(22)   % if p lies outside gamma (other side)
649     Outs4 = b(22);
650     Out15 = In17 -b(22);
651     end
652     end
653     % Junction with 1 incomming and 2 outgoing (4 junctions).
654     %Junction (5 -> s1,6)
655     In6  = min(alpha1*b(5),a(6));
656     Ins1 = min(alpha2*b(5),a(19));
657     Out5 = In6 + Ins1;
658     %Junction (9 -> s3,10)
659     In10  = min(alpha1*b(9),a(10));
660     Ins3  = min(alpha2*b(9),a(21));
661     Out9  = In10 + Ins3;
662     %Junction(13 -> s4,14)
663     In14  = min(alpha1*b(13),a(14));
664     Ins4  = min(alpha2*b(13),a(22));
665     Out13 = In14 + Ins4;
666     %Junction(18 -> s2,16)
667     In16  = min(alpha1*b(18),a(16));
668     Ins2  = min(alpha2*b(18),a(20));
669     Out18 = In16 + Ins2;
670     %-------------- Junction with 6 roads (2 junctions)-----------
671     %Junction (1,6,8 --> 3,4,7)
672     flux1= A*[b(1) ; b(6) ; b(8)];
673     In3 = min(flux1(1),a(3));
674     In7 = min(flux1(2),a(7));
675     In4 = min(flux1(3),a(4));
676     if flux1(1) > a(3)
677         v(1,1) = flux1(1)-a(3);
678     end
679     if flux1(2) > a(7)
680         v(2,1) = flux1(2)-a(7);
681     end
682     if flux1(3) > a(4)
683         v(3,1) = flux1(3)-a(4);
684     end
685     %Right of way parameters(Using matrix A)
686     Out1 = b(1)-A(1,:)*v;
687     Out6 = b(6)-A(2,:)*v;
688     Out8 = b(8)-A(3,:)*v;
689     %Junction (10,14,16 --> 8,12,15)
690     flux2 = A*[b(10) ; b(14) ; b(16)];
691     In8  = min(flux2(1),a(8));
692     In15 = min(flux2(2),a(15));
693     In12 = min(flux2(3),a(12));
694     if flux2(1) > a(8)
695         v2(1,1) = flux2(1)-a(8);
696     end
697     if flux2(2) > a(15)
698         v2(2,1) = flux2(2)-a(15);
699     end
700     if flux2(3) > a(12)
701         v2(3,1) = flux2(3)-a(12);
702     end
```

```
    %Right of way parameters(Using matrix A)
    Out10 = b(10)-A(1,:)*v2;
    Out14 = b(14)-A(2,:)*v2;
    Out16 = b(16)-A(3,:)*v2;
    %--Boundaries--
    %Incomming roads [F1, F5, F13, F18]
    F1(i+1,1) = F1(i,1)  -(k/h)*((U1(i,1)-U1(i,1)^2)-(U01-(U01)^2));
    F5(i+1,1) = F5(i,1)  -(k/h)*((U5(i,1)-U5(i,1)^2)-(U05-(U05)^2));
    F13(i+1,1) = F13(i,1)  -(k/h)*((U13(i,1)-U13(i,1)^2)-(U013-(U013)^2));
    F18(i+1,1) = F18(i,1)  -(k/h)*((U18(i,1)-U18(i,1)^2)-(U018-(U018)^2));
    %Roads with outgoing fluxes [All except: F2, F4, F11, F17]
    F1(i+1,Nx)  = F1(i,Nx)   -(k/h)*(Out1-(U1(i,Nx-1)-U1(i,Nx-1)^2));
    F3(i+1,Nx)  = F3(i,Nx)   -(k/h)*(Out3-(U3(i,Nx-1)-U3(i,Nx-1)^2));
    F5(i+1,Nx)  = F5(i,Nx)   -(k/h)*(Out5-(U5(i,Nx-1)-U5(i,Nx-1)^2));
    F6(i+1,Nx)  = F6(i,Nx)   -(k/h)*(Out6-(U6(i,Nx-1)-U6(i,Nx-1)^2));
    F7(i+1,Nx)  = F7(i,Nx)   -(k/h)*(Out7-(U7(i,Nx-1)-U7(i,Nx-1)^2));
    F8(i+1,Nx)  = F8(i,Nx)   -(k/h)*(Out8-(U8(i,Nx-1)-U8(i,Nx-1)^2));
    F9(i+1,Nx)  = F9(i,Nx)   -(k/h)*(Out9-(U9(i,Nx-1)-U9(i,Nx-1)^2));
    F10(i+1,Nx) = F10(i,Nx) -(k/h)*(Out10-(U10(i,Nx-1)-U10(i,Nx-1)^2));
    F12(i+1,Nx) = F12(i,Nx) -(k/h)*(Out12-(U12(i,Nx-1)-U12(i,Nx-1)^2));
    F13(i+1,Nx) = F13(i,Nx) -(k/h)*(Out13-(U13(i,Nx-1)-U13(i,Nx-1)^2));
    F14(i+1,Nx) = F14(i,Nx) -(k/h)*(Out14-(U14(i,Nx-1)-U14(i,Nx-1)^2));
    F15(i+1,Nx) = F15(i,Nx) -(k/h)*(Out15-(U15(i,Nx-1)-U15(i,Nx-1)^2));
    F16(i+1,Nx) = F16(i,Nx) -(k/h)*(Out16-(U16(i,Nx-1)-U16(i,Nx-1)^2));
    F18(i+1,Nx) = F18(i,Nx) -(k/h)*(Out18-(U18(i,Nx-1)-U18(i,Nx-1)^2));
    Fs1(i+1,Nx) = Fs1(i,Nx) -(k/h)*(Outs1-(Us1(i,Nx-1)-2*Us1(i,Nx-1)^2));
    Fs2(i+1,Nx) = Fs2(i,Nx) -(k/h)*(Outs2-(Us2(i,Nx-1)-2*Us2(i,Nx-1)^2));
    Fs3(i+1,Nx) = Fs3(i,Nx) -(k/h)*(Outs3-(Us3(i,Nx-1)-2*Us3(i,Nx-1)^2));
    Fs4(i+1,Nx) = Fs4(i,Nx) -(k/h)*(Outs4-(Us4(i,Nx-1)-2*Us4(i,Nx-1)^2));
    %Roads with incomming fluxes [All except 1,5,13,18]
    F2(i+1,1)   = F2(i,1)   -(k/h)*((U2(i,1)-U2(i,1)^2)-In2);
    F3(i+1,1)   = F3(i,1)   -(k/h)*((U3(i,1)-U3(i,1)^2)-In3);
    F4(i+1,1)   = F4(i,1)   -(k/h)*((U4(i,1)-U4(i,1)^2)-In4);
    F6(i+1,1)   = F6(i,1)   -(k/h)*((U6(i,1)-U6(i,1)^2)-In6);
    F7(i+1,1)   = F7(i,1)   -(k/h)*((U7(i,1)-U7(i,1)^2)-In7);
    F8(i+1,1)   = F8(i,1)   -(k/h)*((U8(i,1)-U8(i,1)^2)-In8);
    F9(i+1,1)   = F9(i,1)   -(k/h)*((U9(i,1)-U9(i,1)^2)-In9);
    F10(i+1,1)  = F10(i,1) -(k/h)*((U10(i,1)-U10(i,1)^2)-In10);
    F11(i+1,1)  = F11(i,1) -(k/h)*((U11(i,1)-U11(i,1)^2)-In11);
    F12(i+1,1)  = F12(i,1) -(k/h)*((U12(i,1)-U12(i,1)^2)-In12);
    F14(i+1,1)  = F14(i,1) -(k/h)*((U14(i,1)-U14(i,1)^2)-In14);
    F15(i+1,1)  = F15(i,1) -(k/h)*((U15(i,1)-U15(i,1)^2)-In15);
    F16(i+1,1)  = F16(i,1) -(k/h)*((U16(i,1)-U16(i,1)^2)-In16);
    F17(i+1,1)  = F17(i,1) -(k/h)*((U17(i,1)-U17(i,1)^2)-In17);
    Fs1(i+1,1)  = Fs1(i,1) -(k/h)*((Us1(i,1)-2*Us1(i,1)^2)-Ins1);
    Fs2(i+1,1)  = Fs2(i,1) -(k/h)*((Us2(i,1)-2*Us2(i,1)^2)-Ins2);
    Fs3(i+1,1)  = Fs3(i,1) -(k/h)*((Us3(i,1)-2*Us3(i,1)^2)-Ins3);
    Fs4(i+1,1)  = Fs4(i,1) -(k/h)*((Us4(i,1)-2*Us4(i,1)^2)-Ins4);
    %Outgoing Roads [F2, F4, F11, F17]
    F2(i+1,Nx)  = F2(i+1,Nx-1);
    F4(i+1,Nx)  = F4(i+1,Nx-1);
    F11(i+1,Nx) = F11(i+1,Nx-1);
    F17(i+1,Nx) = F17(i+1,Nx-1);
end
```

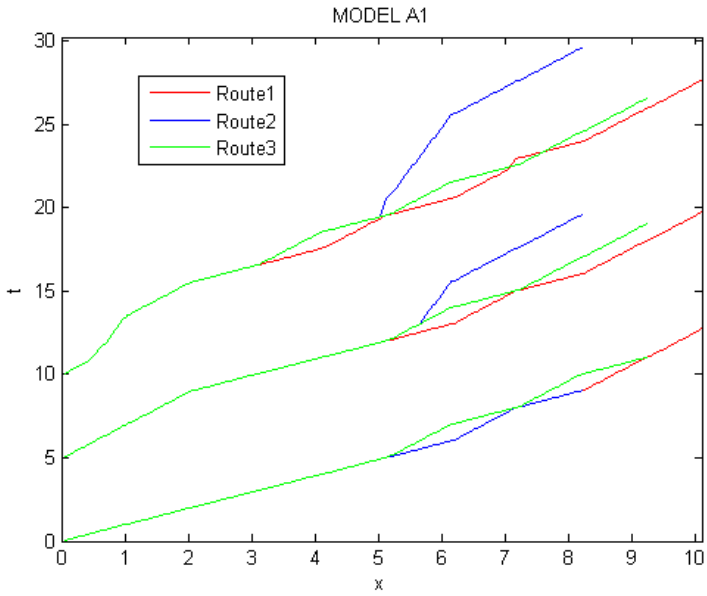

Figure 7.3.6: Plotting time vs position of the different routes. Initial values from test A1.

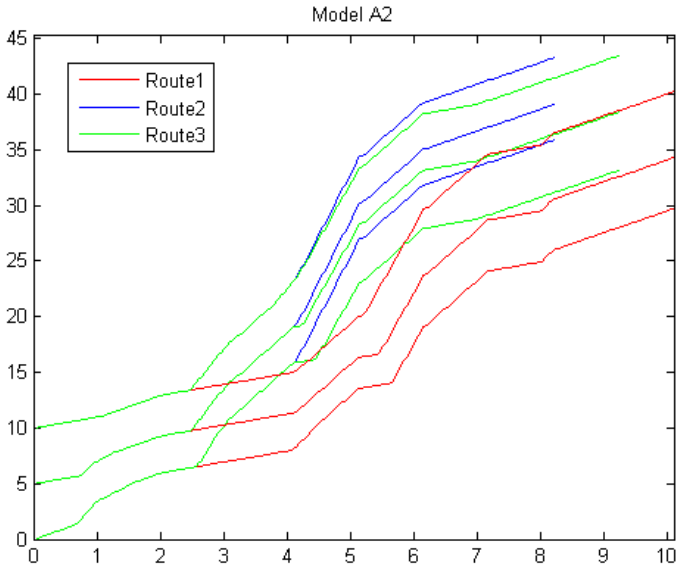

Figure 7.3.7: Plotting time vs position of the different routes. Initial values from test A2.

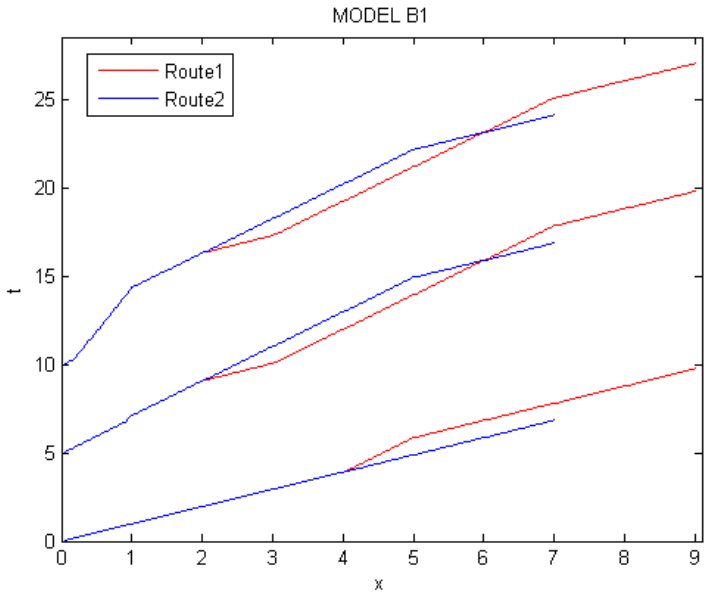

Figure 7.3.8: Plotting time vs position of the different routes. Initial values from test B1.

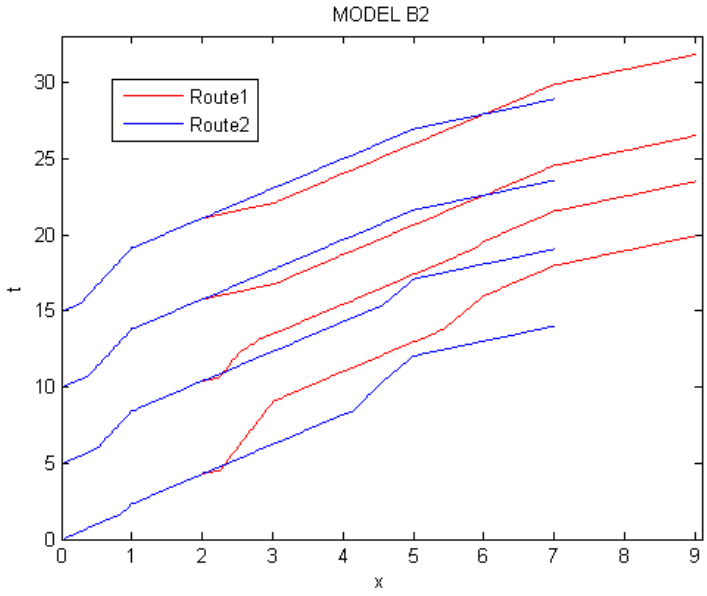

Figure 7.3.9: Plotting time vs position of the different routes. Initial values from test B2.

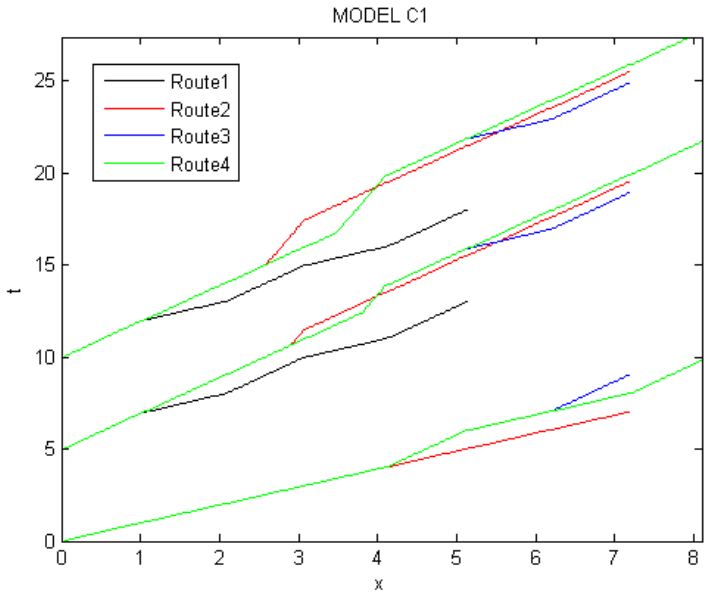

Figure 7.3.10: Plotting time vs position of the different routes. Initial values from test C1.

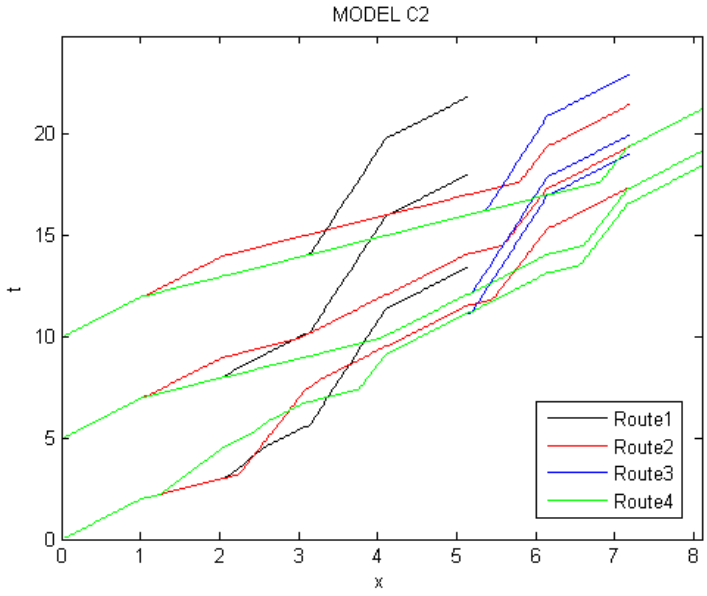

Figure 7.3.11: Plotting time vs position of the different routes. Initial values from test C2.