Estimation of Parameters in Traffic Flow Models Using Data Assimilation

by Amy Jackson

The University of Reading

The Department of Mathematics and Statistics

This dissertation is a joint MSc in the Departments of Mathematics and Meteorology and is submitted in partial fulfilment of the requirements for the degree of Master of Science

September 2011

Acknowledgments

I would like to thank Dr. Amos Lawless and Dr. Peter Sweby for all the help and guidance they have given me.
I would also like to thank my family for their constant encouragement and support throughout this year.

Abstract

The Payne-Whitham model is a macroscopic traffic flow model, usually known as the two equation Payne's model, incorporating two independent parameters denoted by c_0 and τ.

It is implemented by producing an adjoint model, derived from the linearisation of the Harten, Lax and van Leer scheme (HLL).

The purpose of this dissertation is to find the value of these parameters which give the optimal solution when using the model. This involves calculating the cost function, J, and minimising its gradient using data assimilation methods.

The results show that the system was insensitive to τ but demonstrated good resilience for c_0.

Declaration

I confirm that this is my own work and the use of all material from other sources has been properly and fully acknowledged.

Signed ....

Contents

1 Introduction 1
1.1 Background 1
1.2 Overview 1

2 Traffic Flow Models 4
2.1 Choice of the model 4
2.2 Lighthill Whitham Traffic Flow Model 4
2.3 Payne-Whitham Model 5

3 Numerical Schemes- Roe's Algorithm 9
3.1 Introduction 9
3.2 The Theory 9
3.3 Programming Roe 10

4 Numerical Schemes- Harten, Lax and van Leer 12
4.1 Introduction 12
4.2 Scheme 12
4.3 Theory of the HLL Scheme 13
4.4 Results 14

5 Data Assimilation 16
5.1 Four Dimensional Variational Assimilation ($4D-$Var) 17
5.1.1 Useful Notation 17
5.1.2 Error Handling 18
5.2 Generic Cost Function 18
5.3 Cost Function as applied 20
5.4 Minimisation of the Cost Function 21
5.5 Data Assimilation for Parameter Estimation 24

6 Tangent Linear Model **25**
6.1 Introduction 25
6.2 Theory of the TLM 25
6.2.1 25
6.3 Implementation 26
6.3.1 General Example 26
6.3.2 Coding Example 27
6.4 Correctness Testing Results 27

7 Adjoint Model **31**
7.1 Introduction 31
7.2 Theory of the Adjoint Model 31
7.3 Examples 31
7.3.1 General Example 31
7.3.2 Code Implementation Example 32
7.4 Test of the Adjoint Model 33
7.4.1 General Theory of the Validity Test 33

8 Implementing the Minimisation **35**
8.1 Gradient Test 35

9 Results **38**
9.1 Verification 38
9.2 Running the Model 40
9.2.1 Varying c_0 41
9.2.2 Varying τ 42
9.2.3 Varying t 43
9.2.4 Varying σ 45
9.2.5 Further Investigations 46

10 Summary **48**

11 Conclusion 49

12 Appendix A- Glossary 51

13 Appendix B 53

13.1 Data Flow Diagram 53
13.2 Program Outlines 53
13.2.1 Introduction 53
13.2.2 Roes 54
13.2.3 HLL_2files 54
13.2.4 TLM 54
13.2.5 ADJ 55
13.2.6 Merge_err 56
13.2.7 CONMIN 57
13.3 Control file values 57

14 Bibliography 58

1 Introduction

1.1 Background

Much work has been done, particularly over the past thirty years, on traffic flow problems generally based upon a stochastic approach. Just by surfing the web, many studies based on varying theories may be found, among the most popular are those derived from Lighthill Whitham and especially H J Payne.

However, when the traffic density becomes high, the model can be considered as a continuum process (i.e. suitable for motorways or busy roads) but it must be remembered that such theory will not be accurate for low density traffic flow.

The purpose of this dissertation is to further the work begun by Danila Volpi in her dissertation 'Estimation of parameters in traffic flow models using data assimilation' 2009 [1], University of Reading, which was based upon Roe's numerical scheme using a finite difference method. The current dissertation uses an HLL (Harten, Lax van Leer) numerical scheme as an alternative to Roe for numerical modelling of the traffic flow and applies Data Assimilation techniques to estimate the parameters of the model.

1.2 Overview

Traffic problems are a serious global concern with pollution such as experienced in Mexico city, Los Angeles and traffic jams in most urban centres. These problems are set to increase given the forecast in growth of traffic in the coming decades.
Figures from the Department for Transport for the period 1994 - 2010 show a steady overall increasing number of licensed vehicles (excluding motor cycles

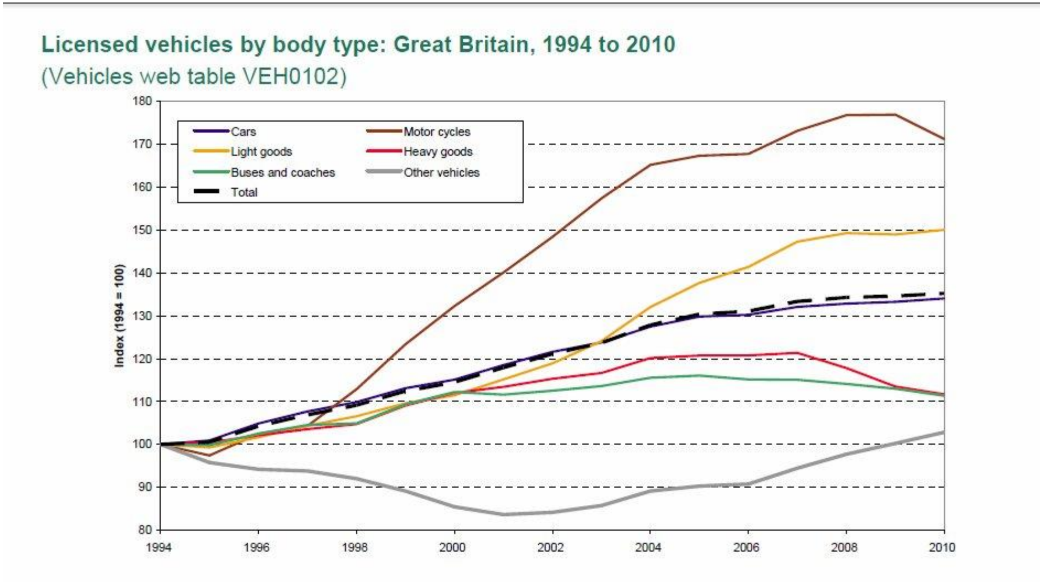


Figure 1: Graph from the Department for Transport [2]

and heavy goods). Although the current economic downturn is reflected in the last two years (see figure 1) the long term prediction is still more vehicles on the road in the UK and across the world generally. One recent estimate suggests an additional 5.7 million cars on the UK roads by 2031, a growth of 21% (Living Streets [3]).

This places increased pressure on the existing road infrastructure in addition to problems caused by pollution and traffic emissions linked to health issues and general atmospheric pollution.

These pollution problems are increased by traffic jams/queues, slow moving traffic which stops and starts and even bus lanes in urban environments where busy roads have their lanes reduced. All this has contributed to an economic loss to the country in terms of reduced output and wasted resources.

This highlights the necessity for optimum road useage to ensure free flowing traffic which will reduce pollution and economic cost towards a minimum.

Traffic modelling is an important part in the solution to these problems as an attempt at understanding the effects of lane reduction on traffic flows and in controlling traffic speeds to provide optimum flow.

This dissertation aims to address these types of problems. It will be implemented using traffic flow modelling techniques that are solved numerically to estimate empirical parameters. These parameters have in the past been chosen by researchers in the light of experience and directly affect the results of the model although they remain essentially constant (in time and space). Four dimensional variational analysis with a data assimilation approach, will be used to obtain improved estimates of these parameters.

2 Traffic Flow Models

2.1 Choice of the model

Apart from having stochastic qualities, three categories of simulation models for traffic have been developed:

- Microscopic models represent individual vehicle movements such as their velocity and position. Although precise, they are computationally very expensive when modelling a large number of vehicles.
- Mesoscopic models represent vehicle movements as groups sometimes referred to as platoons, generally comprising of 3 to 7 vehicles.
- Macroscopic models represent traffic flow in terms of aggregate measures such as density, space-mean-speed, and flow rate. They aim to describe the general system as opposed to each individual vehicle and so are less accurate but more cost effective.

The current document will be concerned with the Macroscopic category, with formulae and concepts originating in meteorology and fluid dynamics being used. The basic formulae are found in a paper titled 'Numerical Simulation of Macroscopic Continuum traffic Models', authors Chin Jian Leo and Robert L. Pretty [4] of the University of Queensland, Australia dated 13 November 1990, to which further references will be made.

2.2 Lighthill Whitham Traffic Flow Model

M. J. Lighthill was a specialist in fluid dynamics, who together with his then PhD student, G. B. Whitham, in 1955 formulated a model, generally known as the LW model, to simulate macroscopic traffic flow.

It is probably the simplest form of a macroscopic traffic flow model comprising of just one equation which treats the traffic as a continuum. The model can be represented essentially by the continuity equation:

$$\rho_t + q_x = 0.$$

By defining $q = \rho v$ we can obtain the equation:

$$\rho_t + (\rho v)_x = 0, \tag{1}$$

where,

- $\rho(x, t)$ is the traffic density (vehicles per km)
- $q(x, t)$ is the traffic flow (vehicles per hour),
- v is the mean traffic speed (km per hour) and has to be prescribed as a function of ρ.

The LW equation (1) allows solutions which were considered too simplistic to represent the required traffic scenario. A better model is given below by Payne.

2.3 Payne-Whitham Model

Limitations were soon found with the LW model, mainly because there was only one state variable, traffic density, which Payne overcame in 1971 by adding a second differential equation including a mean speed-density relationship and introducing terms taking account of drivers anticipation, traffic flow relaxation and changes in traffic volume (vehicles joining or leaving the vehicle stream).

The LW continuity equation (1) is supplemented with the momentum equation which takes the intricate speed-density relationship in traffic into account which, along with the homogenious equation above, is known as the

2-equation Payne's model,

$$(\rho v)_t + (\rho(v^2 + c_o^2))_x = \frac{\rho(U(\rho) - v)}{\tau}, \tag{2}$$

where,

- $v(x,t)$ is the space mean speed (km per hour),
- τ is the relaxation constant,
- c_0^2 is a positive constant,
- and $U(\rho)$ is the equilibrium speed-density relationship.

In this equation (2) there are two parameters, τ and c_0, which are independent from time and space, but clearly have a direct impact on the system. These are the two parameters with which this paper is concerned and will be endeavouring to estimate their ideal values to produce the most accurate forecast possible.

In matrix form equations (1) and (2) can be written as

$$\begin{pmatrix} \rho \\ (\rho v) \end{pmatrix}_t + \begin{pmatrix} 0 & 1 \\ c_o^2 - v^2 & 2v \end{pmatrix} \begin{pmatrix} \rho \\ (\rho v) \end{pmatrix}_x = \begin{pmatrix} 0 \\ \frac{\rho(U(\rho)-v)}{\tau} \end{pmatrix}, \tag{3}$$

and so the homogenous form i.e. no relaxation is as follows:

$$\mathbf{u}_t + A(\mathbf{u})\mathbf{u}_x = 0, \tag{4}$$

where matrix $A(\mathbf{u}) = \begin{pmatrix} 0 & 1 \\ c_o^2 - v^2 & 2v \end{pmatrix}$ and the right hand term of equation (3) is the source term.

It can be shown that equation (3) is hyperbolic if the matrix $A(\mathbf{u})$ has real eigenvalues and is diagonalisable, or if it's eigenvalues are distinct real (J C Strikwerda [5]).

The eigenvalues of matrix $A(\mathbf{u})$ are:

$$\lambda_1 = v + c_0, \qquad \lambda_2 = v - c_0$$

and their corresponding eigenvectors are:

$$e_1 = \begin{pmatrix} 1 \\ v - c_0 \end{pmatrix}, \qquad e_2 = \begin{pmatrix} 1 \\ v + c_0 \end{pmatrix}.$$

Note that λ_1 and λ_2 are distinct real eigenvalues and hence that the system is hyperbolic. A basic feature of systems with hyperbolic equations is the characteristic curves along which there is a constant velocity. It is when these characteristic curves cross that the solutions break down and because each characteristic has a different velocity the resulting solution comprises of two waves, moving with different velocities. The waves can be either a shock or a rarefaction type.

A shock wave is used to represent a changing discontinuity in the model solution, which cannot be handled by the original partial differential equations.

A rarefaction wave occurs as a result of a discontinuity, it fans out and fills up the x, t plane where the characteristic curves fail to.

Numerical schemes are successfully used to solve hyperbolic systems such as that represented by (3) but it is important that a suitable physical solution is chosen to replicate the discontinuity for the correct wave type. Failure to do so will result in the speed of the discontinuity being wrong and misrepresenting actual events. For example, if there is a traffic light change to red and the scheme fails to capture the shock wave, then it may show vehicles continuing through the red light, which should not, and probably would not, happen in reality.

The numerical schemes considered for use in this project are discussed in sections §3 and §4.

3 Numerical Schemes- Roe's Algorithm

3.1 Introduction

Roe's algorithm (P L Roe [6]) is one of the most basic Riemann approximations. It is a well established scheme for aerodynamics which can be used to solve a simple form of a macroscopic traffic model, the Lighthill-Whitham 1-equation model (LW model). Beginning with this, the LW model is essentially the continuity equation (1),

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

where $\rho(x,t)$ is the traffic density measured in vehicles per km, and ρv is the traffic flux, given in vehicles per hour.

3.2 The Theory

Equation (4) is a continuous function which is very difficult to solve and cannot be done analytically. In order to solve it numerically, Roe linearises the homogenious form in each interval (x_{i-1}, x_i), replacing $A(\mathbf{u})$ by matrices $\tilde{A}(\mathbf{u}_{i-1}, \mathbf{u}_i)$ which, for any adjacent states $\mathbf{u}_L$,$\mathbf{u}_R$, the following three conditions are satisfied:

1. Hyperbolicity: $\tilde{A}(\mathbf{u}_L, \mathbf{u}_R)$ is diagonalisable with real eigenvalues, λ
2. Consistency: $\tilde{A}(\mathbf{u}_L, \mathbf{u}_R) \to A(\mathbf{u})$ as $\mathbf{u}_L$ and $\mathbf{u}_R \to \mathbf{u}$
3. Conservation: $\mathbf{f}(\mathbf{u}_L) - \mathbf{f}(\mathbf{u}_R) = \tilde{A}(\mathbf{u}_L, \mathbf{u}_R)(\mathbf{u}_L - \mathbf{u}_R)$
   where $\mathbf{f}(\mathbf{u})_x = A(\mathbf{u})\mathbf{u}_x$

It is necessary to first obtain an $\tilde{A}$ and then diagonalise it, resulting in two scalar equations, one per eigenvalue.

Further details of the scheme are not given as this will not be the scheme used in this work. However it did provide the starting point used for the HLL scheme and also was the scheme used by Danila Volpi [1] and is included here for comparison.

3.3 Programming Roe

A fortran program was written and using 100 space discretisations, $\Delta x = 0.01$, 200 time intervals with $\Delta t = 2.5.10^{-5}$, $\tau = 5$ and $c_0 = 50$, the following graphs were produced showing the density-space and velocity-space relationships at the final time step.

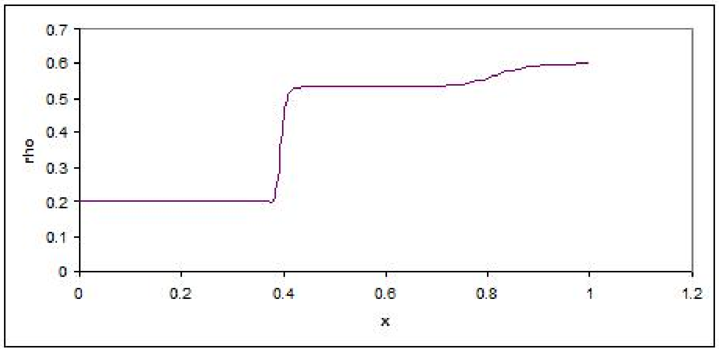


Figure 2: Density-space relationship at the final timestep.

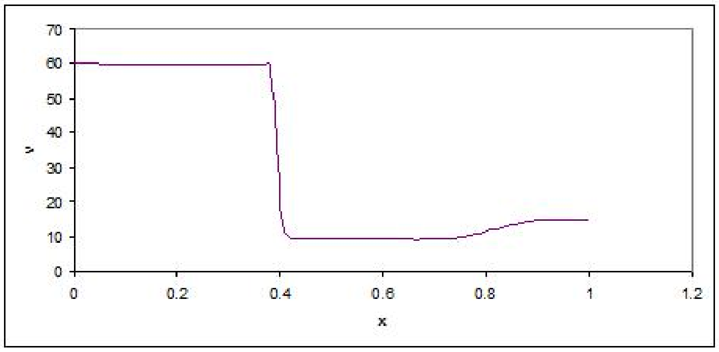


Figure 3: Velocity-space relationship at the final timestep.

A drawback to Roe's method is that is does not check if the solution is entropy satisfying, and so could be incorrect and contain unphysical discontinuities. To deal with this an entropy fix must be added, which will ensure that the solution obtained is correct.

4 Numerical Schemes- Harten, Lax and van Leer

4.1 Introduction

Part of the data assimilation process is to obtain what is known as the tangent linear model which involves linearising the Roe Riemann solver. However Roe has absolute value signs in the numerical flux formula which are difficult to linearise. Another Riemann solver developed by Harten, Lax and van Leer, known as the HLL scheme (Harten et al [7]), does not have this limitation and will be used instead. HLL has the advantage of being a simpler approximation and is a good scheme to use from the conservation law perspective.

The HLL model formulated in 1983 by A. Harten (a PhD student of Lax), P. D. Lax and B. van Leer (specialist in Fluid dynamics) is generally considered to be a more straightforward solution to the Riemann problem, present in the LW and Payne-Whitham models. Although it is an approximate solution, it does provide an efficient computation, a fairly robust result and does not require an entropy fix.

4.2 Scheme

A fortran program was written to perform the following HLL calculations at each value of x within each time step.
They were then incorporated into the HLL scheme defined below:

$$\mathbf{h}_{i-\frac{1}{2}} = \begin{cases} \mathbf{f}_{i-1} & (0 \leq \lambda^L_{i-\frac{1}{2}}) \\ \mathbf{f}^{HLL}_{i-\frac{1}{2}} & (\lambda^L_{i-\frac{1}{2}} \leq 0 \leq \lambda^R_{i-\frac{1}{2}}) \\ \mathbf{f}_i & (\lambda^R_{i-\frac{1}{2}} \leq 0) \end{cases}$$

where

$$\mathbf{f}^{HLL}_{i-\frac{1}{2}} = \frac{\lambda^R_{i-\frac{1}{2}}\mathbf{f}_{i-1} - \lambda^L_{i-\frac{1}{2}}\mathbf{f}_i + \lambda^L_{i-\frac{1}{2}}\lambda^R_{i-\frac{1}{2}}(\mathbf{u}_i - \mathbf{u}_{i-1})}{\lambda^R_{i-\frac{1}{2}} - \lambda^L_{i-\frac{1}{2}}}$$

and were used for the calculation of the state vector **u**:

$$u(j,i) = u(j,i) - \left(\frac{\Delta t}{\Delta x}\right) \cdot (h(j,i+1) - h(j,i)) + \Delta t \cdot s(j,i),$$

where j is the state vector index.

4.3 Theory of the HLL Scheme

The HLL scheme is used to solve equation (3), as described below.

$$\mathbf{u} = \begin{pmatrix} \rho \\ \rho v \end{pmatrix}$$

$$\lambda_L = \tilde{v} - c_0, \qquad \lambda_R = \tilde{v} + c_0$$

where $\tilde{v} = \frac{\sqrt{\rho_L}.v_L + \sqrt{\rho_R}.v_R}{\sqrt{\rho_L}+\sqrt{\rho_R}}$

$$\mathbf{f} = \begin{pmatrix} f_1 \\ f_2 \end{pmatrix} = \begin{pmatrix} \rho v \\ \rho(v^2 + c_0^2) \end{pmatrix}$$

$$f_{L1} = (\rho v)_{i-1}, \qquad f_{R1} = (\rho v)_i$$

$$f_{L2} = \rho_L \cdot (v_L^2 + c_0^2), \qquad f_{R2} = \rho_R \cdot (v_R^2 + c_0^2)$$

$$\mathbf{s} = \begin{pmatrix} 0 \\ \rho_i . \frac{\left(U_{cap} - \frac{(\rho v)_i}{\rho_i}\right)}{\tau} \end{pmatrix}$$

$$U_{cap} = \tanh\left(\frac{1}{\rho_i} - 2\right) + \tanh(2)$$

Where,

- $\mathbf{u}$ is a state vector
- i represents spacial points from 0 to N along the direction of flow in the computational grid
- ρ is the traffic density
- v is the mean speed
- λ are eigenvalues of the matrix $A(\mathbf{u})$ defined in §2.3
- $\mathbf{f}$ is the traffic flux
- L and R signify the left and right states respectively
- $\mathbf{s}$ is the source term
- U_{cap} is the equilibrium speed-density relationship, as in §2.3.

4.4 Results

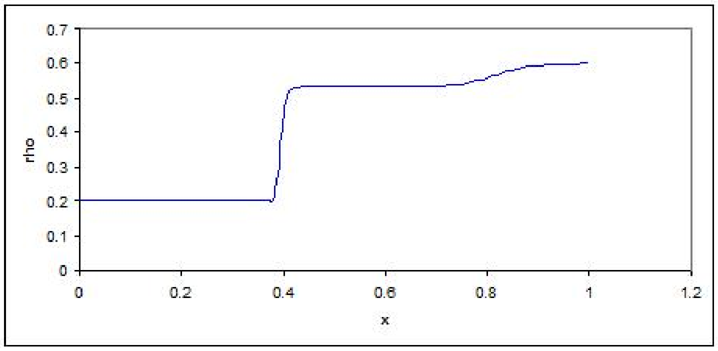


Figure 4: Density-space relationship at the final time step.

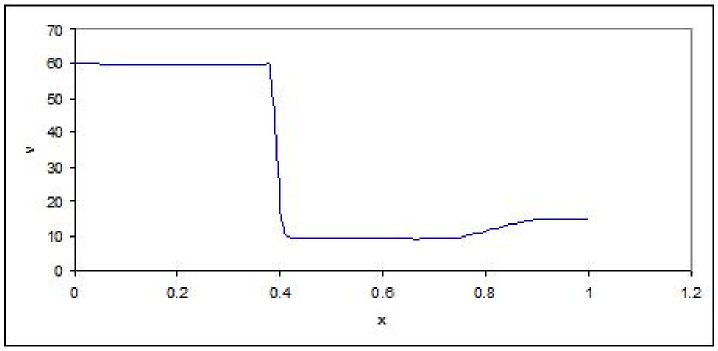


Figure 5: Velocity-space relationship at the final time step.

The graphs produced appear almost identical to those for Roe. Since the HLL method is much simpler and entropy satisfying, no entropy fix is required and so will be suitable for the remaining stages of the project.

5 Data Assimilation

'A data assimilation system consists of three components: a set of observations, a dynamical model, and a data assimilation scheme where the goal is to minimise a cost function with the constraints of the model equations and their parameters.'(A Robinson and P Lermusiaux [8])

It is an iterative process, in two senses, one using time steps where observations are joined with corresponding forecasts from the scheme, (in our case the HLL scheme) for input to the cost function. The other is in the minimisation process where the cost function is minimised to produce the best values for the system parameters forming the parameter vector, **p**.

The approach is based on an augmented state vector, **z**, comprising of the constants c_0 and τ plus the state variables ρ and ρv. That is:

$$\begin{aligned}\mathbf{z} &= (c_0 \quad \tau \quad \rho \quad \rho v)^T \\ &= \begin{pmatrix} \mathbf{p} \\ \mathbf{u} \end{pmatrix}\end{aligned}$$

The data assimilation begins with an initial state x_0, and incorporates observations (current and past) into a numerical model in order to produce a model state, known as the analysis, which most accurately represents the current state of the system.

The model uses the observations in time iterations, i.e. observations across the whole time window can be used (where the time window is given between $(t_0, \ldots, t_n)$). The analysis occurs at t_0 and best represents the actual true state and can be used to make future predictions, such as traffic forecasts for a road closure.

5.1 Four Dimensional Variational Assimilation ($4D-$Var)

$4D-$Var is a model based on the minimisation of an associated cost function which measures the difference between the observations and the forecasts made, weighted by the accuracy of the measurements taken. This project will use $4D-$Var to estimate the state parameters τ and c_0.

5.1.1 Useful Notation

- $3D-$Var, Three-dimensional variational analysis
- $4D-$Var, Four-dimensional variational analysis
- Truth, $\mathbf{x}_T$, The actual true state, e.g. the true temperature of a room
- Analysis, $\mathbf{x}_a$, The analysis is our best estimate of this truth given the information available
- Background, $\mathbf{x}_b$, Prior estimate of the truth before the observations are assimilated, also known as the forecast
- $\mathbf{x}_0$, State at the initial time t_0
- t_k, Time at k^{th} time step
- $\mathbf{x}^k$, State at time t_k
- Observations, $\mathbf{y}$, Observations over a time interval (t_0, t_k)
- Cost function, J, A function of the observation and the background measurements that is minimised to obtain the best estimate of the truth

The reason it is called $4D-$Var is because it incorporates the three spatial dimensions with time. Without incorporating time, this would be $3D$ variational assimilation which only uses each observation once at the time it occurs and then discards it. (N K Nichols [9])

5.1.2 Error Handling

Errors undoubtedly arise in observations which can be caused by numerous reasons, for example inaccuracies in taking the measurements or as a result of using faulty apparatus. If the errors are not dealt with then this will result in an unreliable analysis and so all errors must be minimised for the final analysis.
Observation errors are often correlated especially if they are the result of measurements taken using the same apparatus. These error correlations can be represented in error covariance matrices for calculation purposes. The background error covariance matrix B (defined only for t_0), and the observation error covariance matrix R are known but can be hard to produce when they are formed from multiple sources.
With a large number of observations these matrices are very expensive to invert as B is of size $n \cdot n$ and R is of size $p \cdot p$ where n is the size/length of the state vector and p is the number of observations. (E Kalnay [10])

5.2 Generic Cost Function

As previously mentioned, $4D-$Var is defined as the minimisation of the following cost function which measures the difference between the observations and the forecasts, weighted by the accuracy of the measurements, with the general formula given by:

$$J(\mathbf{x}) = (\mathbf{x} - \mathbf{x}_b)^T B^{-1} (\mathbf{x} - \mathbf{x}_b) + (\mathbf{y} - \mathbf{h}(\mathbf{x}))^T R^{-1} (\mathbf{y} - \mathbf{h}(\mathbf{x})). \qquad (5)$$

So our analysis can be written as:

$$\mathbf{x}_a = min_{\mathbf{x}} J(\mathbf{x}).$$

In this project the only measurements taken are for the state vector variables, i.e. the velocity and the vehicle density, with the equivalent background term excluded as is explained in §5.3. Hence the cost function is given by:

$$J(\mathbf{p}) = \sum_{i=0}^{n} (\mathbf{y}_i - \mathbf{h}_i[\mathbf{x}_i])^T R_i^{-1} (\mathbf{y}_i - \mathbf{h}_i[\mathbf{x}_i])$$

where $\mathbf{h}_i$ is the observation operator.

The cost function ensures that the analysis does not deviate too far from the observations and forecasts which are usually quite reliable. By definition, the cost function is subject to the strong constraint:

$$\mathbf{x}^{(k)} = \mathbf{m}(t_k; t_0; \mathbf{p}; \mathbf{x}_0) \qquad \forall k, \tag{6}$$

where $\mathbf{m}(t_k; t_0; \mathbf{p}; \mathbf{x}_0)$ is the function starting from the initial time, t_0, to t_k.

For the first time step, $k = 0$, $\mathbf{x}^{(0)}$ is the value of the initial state and it is then used to update the forecast using the iterative equation:

$$\mathbf{x}_a = \mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \delta\mathbf{x}^{(k)}, \tag{7}$$

where k goes from $(0, 1, ..., r)$, r is the maximum number of time steps.
$\delta\mathbf{x}^{(k)}$ is an increment of $\mathbf{x}^{(k)}$.

The procedure is repeated until a specified number of time steps have been performed or certain convergence criterion are satisfied (A Lawless, S Gratton, N Nichols [11]). We expect the forecast to produce estimated values for τ and c_0 that give a better representation of the system on succeeding runs.

The final value of $\mathbf{x}$ is defined as $\mathbf{x}_a$.

5.3 Cost Function as applied

For simplicity the background term of the cost function has been removed in the application of this dissertation. The removal is permitted if enough observations exist, although it is also sensible to do where the background values are not known sufficiently accurate. Time permitting, we could experiment with including background values, however, as is said it is simpler to start without it.

Hence, the cost function (5) becomes:

$$\begin{aligned} J(\mathbf{p}) &= \sum_{i=0}^{n} (\mathbf{y}_i - \mathbf{h}_i[\mathbf{x}_i])^T R_i^{-1} (\mathbf{y}_i - \mathbf{h}_i[\mathbf{x}_i]) \\ &= \sum_{i=0}^{n} (\mathbf{y}_i - \mathbf{h}_i[\mathbf{x}_i])^2 R_i^{-1} \\ &= \sum_{i=0}^{n} \left[(\rho_i^o - \rho_i)^2 R_\rho^{-1} + (v_i^o - v_i)^2 R_v^{-1} \right] \end{aligned}$$

where

- $\mathbf{y}_i$ is the observed value of $\mathbf{x}_i$
- $\mathbf{h}_i$ is the observation operator
- $\mathbf{x}_i \equiv \begin{pmatrix} \rho_i \\ v_i \end{pmatrix}$
- ρ_i is the value of ρ at position i
- v_i is the value of v at position i
- ρ_i^o is the observed value of ρ at position i
- v_i^o is the observed value of v at position i

- R_ρ^{-1} is the observation error covariance matrix for ρ
- R_v^{-1} is the observation error covariance matrix for v

This adaption of the cost function is calculated in the HLL scheme.

5.4 Minimisation of the Cost Function

The minimisation of the cost function is the method used to solve the parameter estimation problem, i.e. to find the values of the parameters c_0 and τ which produce the least misfit between the forecast and observational values.

To achieve the minimisation of the cost function, the adjoint model was expanded to obtain the gradient of the cost function. The values produced by this process were passed to the computer package CONMIN (D F Shanno, K H Phua [12]) which calculated the actual minimisation values.

The gradient of the cost function is required with respect to the initial parameter values, $\mathbf{p}_0$, and is given by:

$$\nabla J(\mathbf{p}_0) = -2\sum_{i=0}^{n}\left(\frac{\partial \mathbf{x}_i}{\partial \mathbf{p}_0}\right)^T H_i^T R^{-1}(\mathbf{y}_i - \mathbf{h}_i(\mathbf{x}_i)) \qquad (8)$$

where,

$$\begin{aligned}
\left(\frac{\partial \mathbf{x}_i}{\partial \mathbf{p}_0}\right) &= \left(\frac{\partial(\mathbf{m}_{i-1}(\mathbf{x}_{i-1}))}{\partial \mathbf{p}_0}\right)\\
&= \left(\frac{\partial(\mathbf{m}_{i-1}(\mathbf{x}_{i-1}))}{\partial \mathbf{p}_{i-1}}\right)\left(\frac{\partial(\mathbf{m}_{i-2}(\mathbf{x}_{i-2}))}{\partial \mathbf{p}_0}\right)\\
&= \left(\frac{\partial(\mathbf{m}_{i-1})}{\partial \mathbf{p}_{i-1}}\right)\left(\frac{\partial(\mathbf{m}_{i-2})}{\partial \mathbf{p}_{i-2}}\right)\left(\frac{\partial(\mathbf{m}_{i-3}(\mathbf{x}_{i-3}))}{\partial \mathbf{p}_0}\right)\\
&= \left(\frac{\partial(\mathbf{m}_{i-1})}{\partial \mathbf{p}_{i-1}}\right)\left(\frac{\partial(\mathbf{m}_{i-2})}{\partial \mathbf{p}_{i-2}}\right)...\left(\frac{\partial(\mathbf{m}_0)}{\partial \mathbf{p}_0}\right)\\
&= M_{i-1}M_{i-2}...M_0\\
&= M.
\end{aligned}$$

So equation (8) becomes:

$$\begin{aligned}
\nabla J(\mathbf{p}) &= -2\sum_{i=0}^{n}(M_{i-1}M_{i-2}...M_0)^T H_i^T R^{-1}(\mathbf{y}_i - \mathbf{h}_i(\mathbf{x}_i))\\
&= -2\sum_{i=0}^{n} M^T H_i^T R^{-1}(\mathbf{y}_i - \mathbf{h}_i(\mathbf{x}_i))
\end{aligned}$$

where

- $\mathbf{p} = \begin{pmatrix} c_0 \\ \tau \end{pmatrix}$, the parameter vector,
- M is the linearisation of the non-linear model $\mathbf{m}$,
- H is the Jacobian of the observation operator $\mathbf{h}$,
- $(M_{i-1}M_{i-2}...M_0)^T$ is given by the adjoint model.

For further details on the adjoint model, see section §7.

Perturbations δc_0 and $\delta\tau$ of c_0 and τ are allowed where δc_0 and $\delta\tau$ are $\epsilon\%$ $(1 \leq |\epsilon| \leq 10)$ of the initial values assigned to c_0 and τ respectively. The new values $(c_0 + \delta c_0)$ and $(\tau + \delta\tau)$ are used by the HLL scheme to calculate a new value for the cost function $J(\mathbf{p})$, $\mathbf{p} = \begin{pmatrix} c_0 + \delta c_0 \\ \tau + \delta\tau \end{pmatrix}$.

The same values of the perturbations are used in the adjoint model to calculate the gradient of the cost function evaluated at τ and c_0 using the new values of the perturbations δc_0 and $\delta\tau$.

For example,

$$\delta\tau^n = \left.\frac{\partial J}{\partial \tau}\right|_{t=n}$$

represents the gradient of J with respect to τ at time n.

It is this gradient and the corresponding cost function which are used in the minimisation to obtain the optimal parameters values.

5.5 Data Assimilation for Parameter Estimation

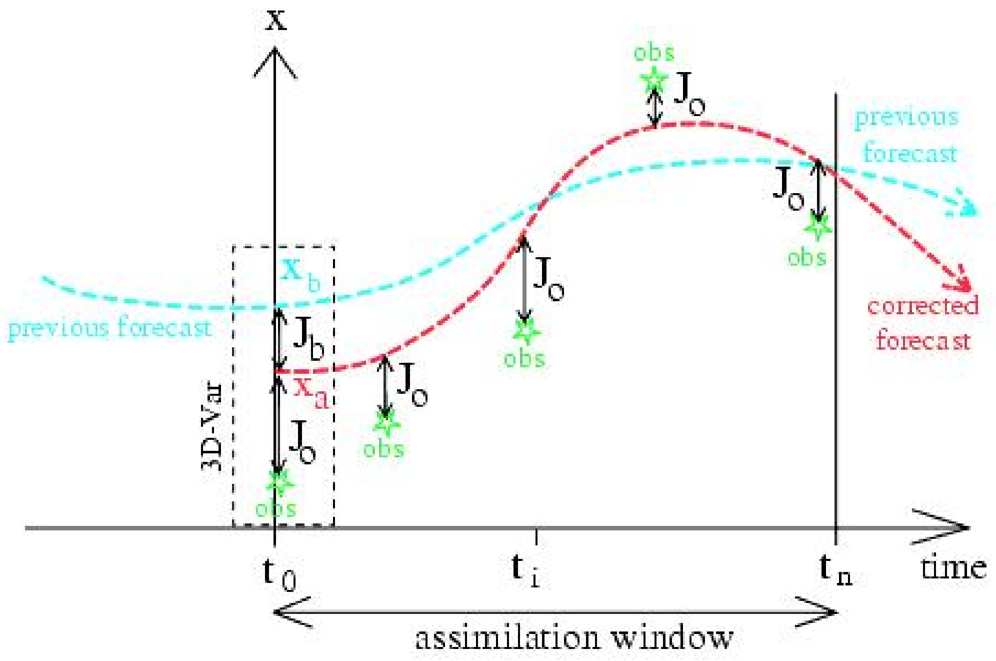


Figure 6: Example of $4D-$Var assimilation in a numerical forecasting system, graph from [13].

In the graphical representation of the $4D-$Var approach, Figure 6, the blue line is the previous forecast and the red is the corrected forecast after the model has been run. It shows that the model is minimising the distance between the forecast trajectory and the observations. This distance is measured by the cost function, $J(\mathbf{p})$, which calculates the weighted sum of the squares of these distances. $4D-$Var is used re-iteratively to minimise the cost function with respect to $\mathbf{p}$, the parameter vector.

By finding the minimum value of this sum we are able to obtain the required values for τ and c_0.

6 Tangent Linear Model

6.1 Introduction

The Tangent Linear Model is essentially the Jacobian of the nonlinear model operator and is therefore derived directly from the HLL model (A Lawless [11]) by keeping the statements of variables themselves unchanged and then adding the related derivatives of these statements. It is required as an intermediate stage to obtain the adjoint model, however note that any logical comparisons remain as in the HLL model. This stage involved the linearisation of the HLL model, for further background theoretical details see N Nichols [14].

6.2 Theory of the TLM

6.2.1

A typical linearisation model is of the form

$$\begin{aligned} \mathbf{x}_i &= \mathbf{m}_i(\mathbf{x}_{i-1}) \\ &= \mathbf{m}_{i-1}\mathbf{m}_{i-2}...\mathbf{m}_0(\mathbf{x}_0) \\ &= \mathbf{m}(\mathbf{x}_0, t_i, t_0) \end{aligned}$$

where $\mathbf{m}$ is the non-linear model and $\mathbf{x}_0$ is the state at the initial time t_0 (E Kalnay [15]).
If the model has initial parameters represented by the vector $\mathbf{p}$, then

$$\mathbf{x}_i = \mathbf{m}(\mathbf{x}_0, \mathbf{p}, t_i, t_0),$$

where $\mathbf{p} = \begin{pmatrix} c_0 \\ \tau \end{pmatrix}$ in this project.

This project is concerned with using observations to make forecasts of future positions and what appear to be random fluctuations in the constants, i.e. the relaxation and anticipation constants when observing real traffic events. This is achieved by including a randomly generated value for $\delta\mathbf{p}$ in the model. Hence,

$$\mathbf{x}_i + \delta\mathbf{x}_i = \mathbf{m}(\mathbf{x}_0, \mathbf{p} + \delta\mathbf{p}, t_i, t_0). \tag{9}$$

Applying a Taylor series expansion to (9) gives:

$\mathbf{x}_i + \delta\mathbf{x}_i = \mathbf{m}(\mathbf{x}_0, \mathbf{p}, t_i, t_0) + \frac{\delta\mathbf{m}}{\delta\mathbf{p}}\delta\mathbf{p} +$ higher order terms

$\Rightarrow \delta\mathbf{x}_i \approx \frac{\delta\mathbf{m}}{\delta\mathbf{p}}\delta\mathbf{p}$, where $\frac{\delta\mathbf{m}}{\delta\mathbf{p}}$ is the TLM represented by $M(\mathbf{x})$.

It should be noted that this approximation holds best if $\mathbf{m}$ is linear, as higher order terms are zero, while non-linear $\mathbf{m}$ will produce higher order terms in $\delta\mathbf{p}^2, \delta\mathbf{p}^3$... resulting in less reliable results. The less linear $\mathbf{m}$ is, the less accurate the results are likely to be.

6.3 Implementation

A fortran program was written to perform these calculations for each value of $\mathbf{x}$ for each time step.

6.3.1 General Example

The tangent linear model was obtained by linearising the nonlinear forward in time HLL code line by line. To code this, every line with a variable in it must be differentiated with respect to each variable in that line.

More explicitly, any line of code in the HLL can be written as:

$$Z = f(\mathbf{X})$$

where

$$\mathbf{X} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$$

where x_1 to x_n are the variables required to obtain Z.

Then the tangent linear code for this is:

$$\delta Z = \left(\frac{\partial f}{\partial x_1}\right)\delta x_1 + \left(\frac{\partial f}{\partial x_2}\right)\delta x_2 + ... + \left(\frac{\partial f}{\partial x_n}\right)\delta x_n.$$

(W Yang and M Navon [16])

6.3.2 Coding Example

In the HLL model code, u is given by:

$$u(j,i) = u(j,i) - \left(\frac{\Delta t}{\Delta x}\right)\cdot(h(j,i+1) - h(j,i)) + \Delta t \cdot s(j,i).$$

The corresponding TLM code is:

$$\delta u(j,i) = \delta u(j,i) - \left(\frac{\Delta t}{\Delta x}\right)\cdot(\delta h(j,i+1) - \delta h(j,i)) + \Delta t \cdot \delta s(j,i).$$

It is important to also retain the original lines of HLL code as some of them will still be needed in calculations.

6.4 Correctness Testing Results

Human error is highly likely to occur when doing such code manipulations as it is so easy to make a mistake. There exists a test for the correctness of the TLM which involves calculating and plotting the relative errors for ρ and

v. Both plots should tend to zero.

The non-linear model (HLL) is run twice, once with unperturbed values for τ and c_0 giving $m(\mathbf{p})$, and once with perturbed values for τ and c_0, giving $m(\mathbf{p} + \delta\mathbf{p})$. The perturbation vector is given by $\delta\mathbf{p} = \begin{pmatrix} \delta c_0 \\ \delta\tau \end{pmatrix}$. Following this, the tangent linear model is run once with the perturbed τ and c_0 to give $M\delta\mathbf{p}$.

The total perturbation, $m(\mathbf{p}+\delta\mathbf{p}) - m(\mathbf{p})$, is then compared with it's linear component, $M\delta\mathbf{p}$. (Y Li et al [17]) This is performed by calculating the relative error as shown below.

The relative error is given by:

$$\frac{\|m(\mathbf{p}+\delta\mathbf{p}) - m(\mathbf{p}) - M\delta\mathbf{p}\|}{\|M\delta\mathbf{p}\|} \cdot 100$$

where, $\|\mathbf{x}\| = \sqrt{\sum_{i=1}^{N} x_i^2}$ is the L_2 norm.

Once this was calculated, the logarithmic relative error was plotted against decreasing perturbation sizes, α, where $\alpha = 10^0, 10^{-1}, 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}$ for the velocity plot, and from 10^0 to 10^{-6} for the traffic density plot. The graphs obtained are shown below:

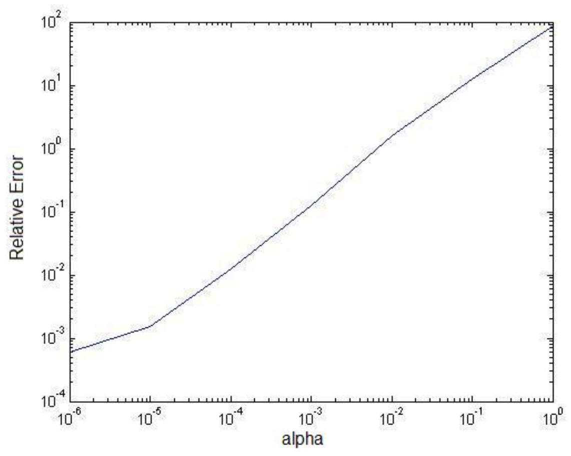


Figure 7: A graph to show the correctness of the TLM after 200 timesteps for traffic density.

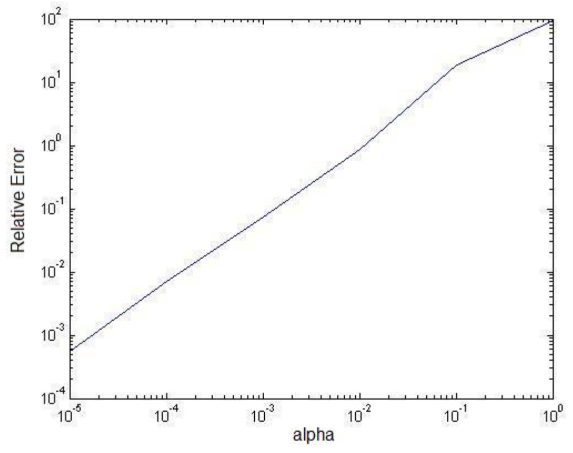


Figure 8: A graph to show the correctness of the TLM after 200 timesteps for velocity.

As can be seen in figures 7 and 8 as α decreases, so do the relative errors. For ρ, the relative error decreases over 6 orders magnitude and over 5 for v (figure 8). Both graphs tend to zero which is the desired result and strongly suggests that the TLM has been done correctly.

It should be noted that for very small perturbation sizes (10^{-8} and smaller), the error began to increase but this was due to rounding errors in the program and does not invalidate the model but is a limitation of the machine.

7 Adjoint Model

7.1 Introduction

The adjoint model produces the gradient of the cost function which is an intrinsic part of the minimisation process.
The adjoint variables represent the gradient of the cost function with respect to the model variables. The TLM starts from the initial HLL values and calculates the variation between the unperturbed and the perturbed values for the tangent linear model at $t = t_{max}$; the adjoint model will start with the final values produced by TLM and run backwards in both space and time to estimate the initial state vector values i.e. at $t = 0$.

7.2 Theory of the Adjoint Model

In order to obtain the adjoint of a linear model, it must be presented in the form $\mathbf{x}^{n+1} = M\mathbf{x}^n$. The adjoint is then $\hat{\mathbf{x}}^n = M^T\hat{\mathbf{x}}^{n+1}$. Adjoint models are very useful for computing the derivatives of a function which has numerous input variables, and so is particulary good for parameter and/or state vector estimation.
The Adjoint model was generated from the TLM model by 'reversing' the logic/code and exchanging the derivatives on either side of statements; where they did not exist, the statements remain unchanged.

7.3 Examples

7.3.1 General Example

For example, take the following linear model of two variables y, z with

$$y^{n+1} = \alpha y^n + \beta z^n$$

$$z^{n+1} = \xi y^n + \zeta z^n.$$

This can be written as $\begin{pmatrix} y \\ z \end{pmatrix}^{n+1} = \begin{pmatrix} \alpha & \beta \\ \xi & \zeta \end{pmatrix} \begin{pmatrix} y \\ z \end{pmatrix}^{n}$,

i.e. $\mathbf{x}^{n+1} = M\mathbf{x}^n$ where $\mathbf{x} = \begin{pmatrix} y \\ z \end{pmatrix}$ and $M = \begin{pmatrix} \alpha & \beta \\ \xi & \zeta \end{pmatrix}$.

Then the adjoint model $\hat{\mathbf{x}}^n = M^T \hat{\mathbf{x}}^{n+1}$ is as follows:

$$\begin{pmatrix} \hat{y} \\ \hat{z} \end{pmatrix}^{n} = \begin{pmatrix} \alpha & \xi \\ \beta & \zeta \end{pmatrix} \begin{pmatrix} \hat{y} \\ \hat{z} \end{pmatrix}^{n+1},$$

i.e.

$$\hat{y}^n = \alpha \hat{y}^{n+1} + \xi \hat{z}^{n+1}$$
$$\hat{z}^n = \beta \hat{y}^{n+1} + \zeta \hat{z}^{n+1}.$$

The adjoint of the TLM is effectively the transpose of the TLM. Implementing this in the program is somewhat complicated as every line of the TLM code with variables had to be altered and split into separate equations.

7.3.2 Code Implementation Example

Using the same example as in §6.3.2, the TLM code δu is given by:

$$\delta u(j,i) = \delta u(j,i) - \left(\frac{\Delta t}{\Delta x}\right) \cdot (\delta h(j,i+1) - \delta h(j,i)) + \Delta t \cdot \delta s(j,i).$$

This can be written as:

$$\begin{pmatrix} \delta u(j,i) \\ \delta h(j,i+1) \\ \delta h(j,i) \\ \delta s \end{pmatrix} = \begin{pmatrix} 1 & -\frac{\Delta t}{\Delta x} & \frac{\Delta t}{\Delta x} & \Delta t \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \delta u(j,i) \\ \delta h(j,i+1) \\ \delta h(j,i) \\ \delta s \end{pmatrix},$$

the adjoint model is:

$$\begin{pmatrix} \delta\hat{u}(j,i) \\ \delta\hat{h}(j,i+1) \\ \delta\hat{h}(j,i) \\ \delta\hat{s} \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ -\frac{\Delta t}{\Delta x} & 1 & 0 & 0 \\ \frac{\Delta t}{\Delta x} & 0 & 1 & 0 \\ \Delta t & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \delta\hat{u}(j,i) \\ \delta\hat{h}(j,i+1) \\ \delta\hat{h}(j,i) \\ \delta\hat{s} \end{pmatrix}.$$

Therefore the corresponding code for the adjoint model of δu (in TLM) is:

$$\begin{aligned} \delta\hat{u}(j,i) &= \delta\hat{u}(j,i) \\ \delta\hat{h}(j,i+1) &= \delta\hat{h}(j,i+1) - \left(\frac{\Delta t}{\Delta x}\right) \cdot (\delta\hat{u}(j,i)) \\ \delta\hat{h}(j,i) &= \delta\hat{h}(j,i) - \left(\frac{\Delta t}{\Delta x}\right) \cdot (-1) \cdot (\delta\hat{u}(j,i)) \\ \delta\hat{s}(j,i) &= \delta\hat{s}(j,i) + \Delta t \cdot \delta\hat{u}(j,i). \end{aligned}$$

After all the timesteps have been completed, it is necessary to set $\delta\hat{u}(j,i)$ equal to zero (Chao and Chang [18]) since the adjoint model is the reverse process of the TLM and therefore should finish with the initial values of the TLM.

7.4 Test of the Adjoint Model

The method of writing the adjoint Fortran code had the consequence that mistakes could easily be made even though great care was taken. Testing was very difficult because the code comprised mainly of lines of calculation so a validity test was performed.

7.4.1 General Theory of the Validity Test

The inner product of two vectors $\mathbf{x}$ and $\mathbf{y}$ is given by:

$$\langle \mathbf{x}, \mathbf{y} \rangle = \sum_{i=0}^{n} x_i y_i = \mathbf{x}^T \mathbf{y}$$

where the triangular brackets denote the inner product.

The procedure for the validity test is as follows, where $\mathbf{M}$ is an operator and $\mathbf{M}^T$ is its adjoint.

1. We begin with a random perturbation $\delta\mathbf{x}$.
2. Then, the TLM code is applied, giving $\mathbf{M}\delta\mathbf{x}$.
3. The adjoint model is then applied to $\mathbf{M}\delta\mathbf{x}$ to obtain $\mathbf{M}^T\mathbf{M}\delta\mathbf{x}$.
4. Calculate $\langle \mathbf{M}\delta\mathbf{x}, \mathbf{M}\delta\mathbf{x} \rangle$.
5. Calculate $\left\langle \delta\mathbf{x}, \mathbf{M}^T\mathbf{M}\delta\mathbf{x} \right\rangle$.
6. Check that $\langle \mathbf{M}\delta\mathbf{x}, \mathbf{M}\delta\mathbf{x} \rangle = \left\langle \delta\mathbf{x}, \mathbf{M}^T\mathbf{M}\delta\mathbf{x} \right\rangle$.

When this test was done, $\langle \mathbf{M}\delta\mathbf{x}, \mathbf{M}\delta\mathbf{x} \rangle$ and $\left\langle \delta\mathbf{x}, \mathbf{M}^T\mathbf{M}\delta\mathbf{x} \right\rangle$ produced the same value, from which it was concluded that the adjoint had been coded correctly. (Lawless et al [19])

8 Implementing the Minimisation

When the adjoint model had been obtained and verified, the gradient of the cost function J was calculated and tested to ensure correctness.
The gradient, ∇J, of the cost function, as determined by the HLL scheme, is calculated within the adjoint model. This value is then used in the gradient test, described in §8.1, to ensure that the adjoint model is working correctly. The gradient, ∇J, is given by equation (8).

The outputs from the adjoint model are $\delta\tau$ and δc_0 where

$$\delta\tau^n = \left.\frac{\partial J}{\partial \tau}\right|_{t=n}, \qquad \delta c_0^n = \left.\frac{\partial J}{\partial c_0}\right|_{t=n}.$$

As explained in §5.4, $\delta\tau^n$ is the gradient of J with respect to τ at time n and δc_0^n the gradient of J with respect to c_0 at time n.
To be confident that these values are the correct gradient calculations of J, the gradient test was applied.

8.1 Gradient Test

The HLL scheme calculates the cost function $J(\mathbf{x}, \mathbf{p})$ for given values of c_0 and τ. It then uses perturbed values of these parameters to produce a perturbed value of J, given by:

$$J(\mathbf{x}, \mathbf{p} + \alpha\delta\mathbf{p}).$$

Using the Taylor expansion:

$$J(\mathbf{x}, \mathbf{p} + \alpha\delta\mathbf{p}) = J(\mathbf{x}, \mathbf{p}) + \alpha\delta\mathbf{p}^T \nabla J(\mathbf{x}, \mathbf{p}) + O(\alpha^2).$$

Rearranging:

$$\frac{J(\mathbf{x}, \mathbf{p} + \alpha\delta\mathbf{p}) - J(\mathbf{x}, \mathbf{p})}{\alpha\delta\mathbf{p}^T \nabla J(\mathbf{x}, \mathbf{p})} = 1 + O(\alpha).$$

Now define

$$\Phi(\alpha) = \frac{J(\mathbf{x}, \mathbf{p} + \alpha\delta\mathbf{p}) - J(\mathbf{x}, \mathbf{p})}{\alpha\delta\mathbf{p}^T \nabla J(\mathbf{x}, \mathbf{p})}$$

where α takes the values $1, 0.1, 0.01, ..., 10^{-11}$ and $\delta\mathbf{p} = \begin{pmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{pmatrix}$.

For values of α that are small, but quite not zero, $\Phi(\alpha)$ should show a constant value close to 1 (Navon et al [20]). Figure 9 clearly shows $\Phi(\alpha) = 1$ over an interval of 6 orders of magnitude. Φ's lower limit is restricted by the accumulation of rounding errors arising in the computer used.

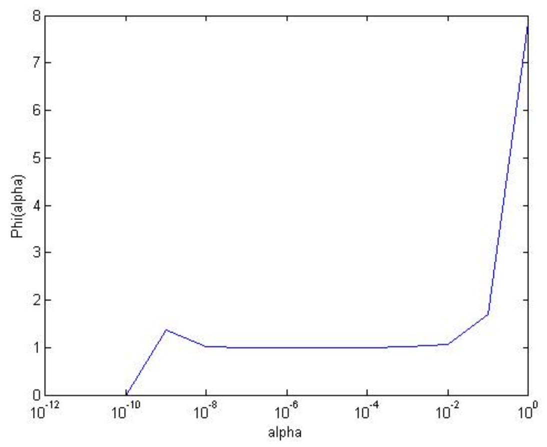


Figure 9: A graph showing $\Phi(\alpha)$ against α.

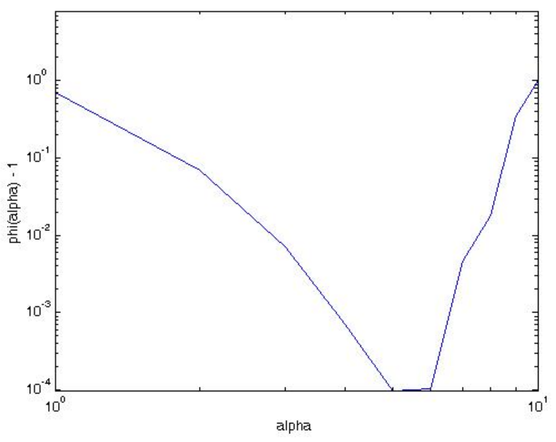


Figure 10: A graph showing $(\Phi(\alpha) - 1)$ against α.

9 Results

The purpose of the project is to determine the optimal values for τ and c_0 which produce the closest match between the forecast and observed measurements.
To do so, the HLL scheme and the adjoint model are combined with the computer module CONMIN to minimise the cost function, J, and the modulus of it's gradient, ∇J, for all specified time steps. To verify that $4D-$Var is suitable to obtain the parameters, it is necessary to investigate the effects of altering the initial 'guess' values of τ and c_0, together with changing the size of the time window, t, and adjusting the amplitude of the noise on the observations, represented by the observation error covariences (σ_ρ and $\sigma_{\rho v}$). For testing purposes, a random number generator was used to generate fluctuations of the u values to produce the observation values used by the cost function.

Although ∇J is one of the factors of the minimisation process, it should be noted that it is not possible to solve exactly for $\nabla J = 0$, which is the reason a user specified convergence attribute, EPS, is embedded within CONMIN.

Note: in CONMIN, EPS imposes convergence when

$$\|\nabla J\| \leq \beta \cdot \max\{1, \|\mathbf{p}\|\},$$

where β is the value of the EPS attribute.

9.1 Verification

The figures below are graphical representations of J and $\|\nabla J\|$ against the CONMIN iteration number. Figure 11 clearly shows J decreasing and then converging to a value after three iterations. Whereas Figure 12 illustrates $\|\nabla J\|$ decreasing steadily and approaching zero after seven iterations. These two graphs demonstrate the minimisation process and affirm that CONMIN has been implemented correctly.

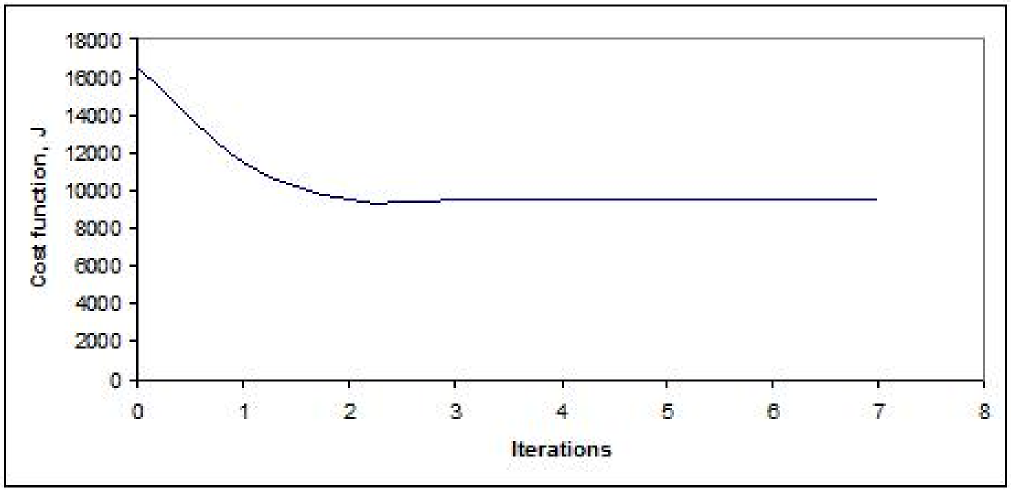


Figure 11: A graph to show J against the iterations.

The data for the graphs was produced using the values:

$$\begin{aligned} c_0^{true} &= 50 \qquad c_0^{guess} = 55 \\ \tau^{true} &= 5 \qquad \tau^{guess} = 4.5 \\ \sigma &= 1 \cdot 10^{-5} \end{aligned}$$

and the time window, $t = 3.75 \cdot 10^{-3}$.

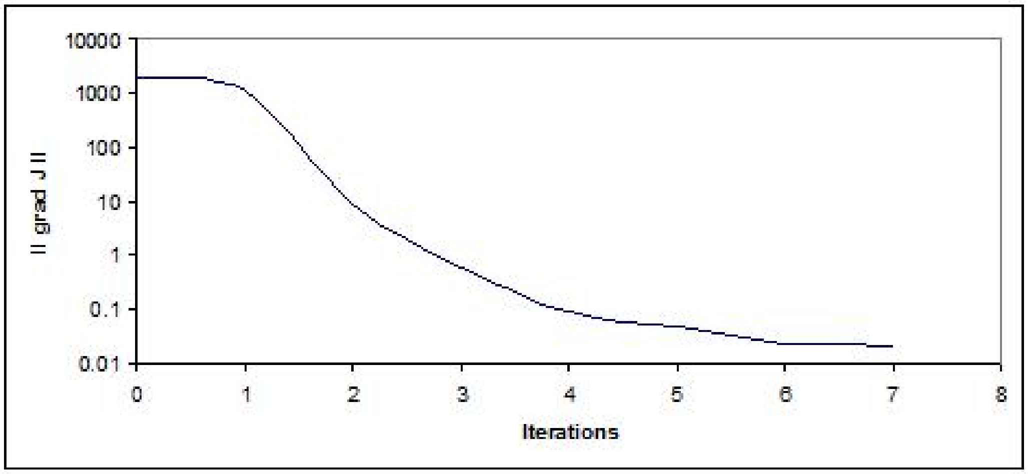


Figure 12: A graph to show $\|\nabla J\|$ against the iterations.

9.2 Running the Model

As mentioned above, the effects of altering $\mathbf{p}^{guess}$, t and σ will be explored to see how well the model can attain the best fit parameters. This must be done systematically so the effect of each change can be seen. For all of the following results the true parameter values were:

$$c_0^{true} = 50, \qquad \tau^{true} = 5.$$

One item of information produced by CONMIN, the number of iterations (ι), merits comment here. This value represents the number of iterations made by CONMIN to obtain the estimates provided by the minimisation routine, or as a result of the CONMIN convergence process. It gives a direct correlation with the processing required to provide a solution from the 'guess' values. Generally this value tended to increase as the time window increased, but even this was not consistent as the results in Table 9 show. Contrary to expectations, large values of the guessed values did not always produce increased number of iteratons, but as can be seen in Table 9, produced quite low iteration numbers.

9.2.1 Varying c_0

In the results below the superscripts ^{g}, ^{e} and ^{t} stand for guess, estimate and true values respectively, and ι is the CONMIN iteration number.
It is expected that when c_0^g and τ^g are given values with a greater difference from their true values, the minimisation will take longer to process, i.e. more iterations will be required to produce a close estimate. If they deviate too far, then the model might not produce close values.

c_0^g	10	20	40	50	55	60	80	100
τ^g	4.5							
t	$5 \cdot 10^{-3}$							
σ	10^{-6}							
c_0^e	50	50	50	50	50	50	50	50
τ^e	5.000	5.000	5.000	5.000	5.000	5.000	5.000	5.000
J	10447.3	10447.3	10447.3	10447.3	10447.3	10447.3	10447.3	10447.3
∇J	1113.6	24162.6	7919.5	932.8	5551.7	3906.1	10738.4	4876.2
ι	20	11	13	13	15	10	14	9

Table 1: c_0 varying with $\sigma = 10^{-6}$

Table 1 shows that varying c_0^g when the observations are almost perfect has little effect on obtaining the optimum values of the parameters as they both approached their true values. This shows that convergence has occurred independent of the original guess value for c_0. It also shows that since the values of ∇J are far from zero, convergence has occurred in CONMIN before the minimisation process was completed, a consequence of the value of the EPS parameter and the small covariance values.

c_0^g	10	20	40	50	55	60	80	100
τ^g	4.5							
t	$5 \cdot 10^{-3}$							
σ	0.7							
c_0^e	49.443	49.443	49.443	49.443	49.443	49.443	49.443	49.443
τ^e	124469.6	190869.8	161542.0	167884.9	188998.3	216759.4	169437.1	136860.9
J	12546.3	12546.3	12546.3	12546.3	12546.3	12546.3	12546.3	12546.3
∇J	0.8	0.6	0.7	0.7	0.6	0.4	0.7	0.8
ι	32	29	27	26	28	33	29	29

Table 2: c_0 varying with $\sigma = 0.7$

This is a general point concerning the results, that the solution may be reached when the specified level of convergence is obtained, irrespective of whether minimisation has been completed.

When the same values are used with less perfect observations, the model still manages to reach a close value for c_0, of 49.443. However this change has caused τ^e to have extremely large values (see Table 2) which do not appear to correlate with the values of c_0. However, J converges to 12546.3 and ∇J tends to zero suggesting that minimisation has occurred.

9.2.2 Varying τ

Varying τ^g had no effect on the value of c_0^e in the tests conducted as can be seen in Tables 3 and 4. In each of these cases, a good value of ∇J that approached zero was recorded, although the values of τ^e were far from the true value especially so for the higher value of σ.
From this, it can be concluded that minimisation was achieved giving a good value of c_0 despite τ varying and hence the process appeared to be highly insensitive to the parameter τ (for reasonable values).

c_0^g	55							
τ^g	3	4	5	6	7	10	15	30
t	$5 \cdot 10^{-3}$							
σ	0.01							
c_0^e	50.01	50.01	50.01	50.01	50.01	50.01	50.01	50.01
τ^e	48.92	48.29	48.29	48.28	48.29	48.29	48.29	48.29
J	10447.8	10447.8	10447.8	10447.8	10447.8	10447.8	10447.8	10447.8
∇J	0.051	0.017	0.045	0.158	0.028	0.005	0.084	0.037
ι	29	24	21	22	22	25	22	16

Table 3: τ varying with $\sigma = 0.01$

c_0^g	55							
τ^g	3	4	5	6	7	10	15	30
t	$5 \cdot 10^{-3}$							
σ	0.7							
c_0^e	49.443	49.443	49.443	49.443	49.443	49.443	49.443	49.443
τ^e	158542	131612	187058	99929	162814	157938	96639	123660
J	12546.3	12546.3	12546.3	12546.3	12546.3	12546.3	12546.3	12546.3
∇J	0.7	0.8	0.7	0.9	0.7	0.7	0.3	0.8
ι	30	27	28	28	27	30	28	24

Table 4: τ varying with $\sigma = 0.7$

9.2.3 Varying t

The project uses a linear model to simulate a non-linear model problem and with a relatively small time window, $t = 1.25 \cdot 10^{-3}$, the problem approximates to a linear solution which will generate meaningful values for J. However with a longer time window, it is less likely that it will be as well approximated by a linear model, because J could have more than one local minimum which were formerly not apparent, but by increasing t they could emerge.

Despite expecting the time interval to have an effect on the results, Table 5 shows that the model attained very good estimates for the parameters, regardless of t. Table 6, with a larger value of σ, produce similar values for c_0 and the cost function, however τ fluctuated widely and the gradient to a lesser extent.

It should be noted that as t increases, since δt remains constant throughout, more observations are used in the cost function and hence in the minimisation calculations.

c_0^g	55							
τ^g	4.5							
t	0.00025	0.00125	0.0025	0.00375	0.005	0.00625	0.0075	0.0125
σ	10^{-6}							
c_0^e	50	50	50	50	50	50	50	50
τ^e	5.007	5.001	5.001	5.001	5.001	5.000	5.000	5.000
J	$4.8 \cdot 10^2$	$2.5 \cdot 10^3$	$5.1 \cdot 10^3$	$7.8 \cdot 10^3$	$1.0 \cdot 10^4$	$1.3 \cdot 10^4$	$1.6 \cdot 10^4$	$2.7 \cdot 10^4$
∇J	2.6	$4.3 \cdot 10^6$	$8.9 \cdot 10$	$1.7 \cdot 10^6$	$5.6 \cdot 10^3$	$1.2 \cdot 10^7$	$1.4 \cdot 10^7$	$4.4 \cdot 10^6$
ι	11	31	11	28	27	18	30	20

Table 5: t varying with all other parameters remaining constant, using $\sigma = 10^{-6}$.

c_0^g	55							
τ^g	4.5							
t	0.00025	0.00125	0.0025	0.00375	0.005	0.00625	0.0075	0.0125
σ	10^{-2}							
c_0^e	49.98	49.99	50.00	50.01	50.01	50.01	50.01	50.01
τ^e	2201185.5	8.9	3898948.0	491.8	50.0	23.5	16.7	11.6
J	$6.4 \cdot 10^2$	$3.1 \cdot 10^3$	$5.3 \cdot 10^3$	$7.8 \cdot 10^3$	$1.0 \cdot 10^4$	$1.3 \cdot 10^4$	$1.6 \cdot 10^4$	$2.7 \cdot 10^4$
∇J	0.83	222.45	13.45	486.30	4.84	643.37	465.09	43.67
ι	27	13	34	35	29	20	20	23

Table 6: t varying with all other parameters remaining constant, using $\sigma = 10^{-2}$.

9.2.4 Varying σ

The impact of varying the observation error covariance matix, $R = \sigma^2 I$, applies directly to the observation values, $\mathbf{y}$. The nearer σ is to zero, the closer the observations are to the true state of $\mathbf{u}$ and it would be expected that the model would produce parameter estimations close to their true values. Conversely, as σ increases, the observations deviate further from $\mathbf{u}$, and the model is likely to yield less close estimates of $\mathbf{p}^t$.

Table 7 supports this reasoning. It can clearly be seen that for $\sigma = 10^{-6}$, $\mathbf{p}^e = \mathbf{p}^t$, but as σ increases, the estimates for both c_0 and τ deteriorate as expected.

The data in this table further supports the statement made previously about convergence occurring prior to minimisation as it can be seen that c_0 has good values inspite of the large values for ∇J.

c_0^g	55							
τ^g	4.5							
t	$5 \cdot 10^{-3}$							
σ	10^{-6}	10^{-4}	10^{-3}	10^{-2}	10^{-1}	10^{0}	2	5
c_0^e	50.000	50.000	50.001	50.009	49.938	49.185	48.237	44.023
τ^e	5.000	5.045	5.493	48.290	1211698	123999.7	23747	163.2
J	10447.3	10447.3	10447.3	10447.8	12158.0	12563.6	12563.6	12425.6
∇J	5551.7	64.3	2.5	0.0	4.6	0.5	4.8	0.4
ι	15	13	14	28	35	28	28	14

Table 7: σ varying with all other parameters remaining constant.

9.2.5 Further Investigations

Further investigations were conducted, some of which are included in Tables 8 and 9 below. The tests involved varying two or more parameters at the same time and using more extreme values for the parameters. This tests the model's resilience when using exaggerated values.

The first observation from Table 8 is that c_0 was estimated accurately irrespective of the values of c_0, τ or σ, within the given ranges. τ also approximated well for small σ even though it's initial value deviated significantly from the true value. However for larger values of σ, as τ^g approached it's true value, the estimated value became highly inaccurate.
This agrees with the previous comments about the insensitivity of τ which receives further support from the results in Table 9, where extreme values of τ^g, providing that $\sigma \leq 1$, did not prevent the attainment of very good values for c_0^e. For higher values of σ (> 1), as σ increased, the values for c_0^g deteriorated despite good values for ∇J and an increasing time window.

c_0^g	80	74	68	62	56	50
τ^g	3	3.4	3.8	4.2	4.6	5
t	$5 \cdot 10^{-3}$					
σ	10^{-6}	10^{-5}	10^{-4}	10^{-3}	10^{-2}	10^{-1}
c_0^e	50.000	50.000	50.000	50.001	50.009	49.938
τ^e	5.000	5.004	5.045	5.493	49.289	1238234
J	10477.3	10477.3	10477.3	10477.3	10477.8	12158.0
∇J	11939.6	116.7	17.5	0.2	0.1	10.5
ι	12	14	15	16	22	31

Table 8: Varying three parameters.

c_0^g	80	80	70	70	55	55	55	101
τ^g	$2 \cdot 10^3$	10^{-1}	$5 \cdot 10^3$	1	10^5	10^5	10^5	10^5
t	0.00625	0.00625	0.00625	0.01	0.01	0.01	0.01	0.005
σ	0.01	0.1	1	2	5	9	0.1	0.1
c_0^e	50.01	49.95	49.26	48.60	44.36	38.43	49.97	49.94
τ^e	$2 \cdot 10^3$	$1.9 \cdot 10^5$	$5 \cdot 10^3$	$8 \cdot 10^2$	10^5	10^5	10^5	10^5
J	13373	15098	15611	21640	23658	22892	23887	12158
∇J	176.16	0	1.02	4.29	0.20	0.09	67.83	0.37
ι	5	44	5	30	4	6	4	3

Table 9: Varying various combinations of the parameters.

10 Summary

The advantage of using $4D-$Var is that it uses each observation in the time window at every iteration, i.e. the observations are treated at the correct time. This makes the forecast produced highly accurate and it incorporates all the available information into the model. However it does therefore require all the observations over the whole time window to be available before the analysis can be calculated which can delay the availability of $\mathbf{x}_a$. When dealing with real data this may become a problem.

Naturally, there are limitations to using the adjoint model due to limits on the predictability of traffic flow. The model can't predict a spontaneous major event, such as a crash, but can only model the effects that this will have (R Errico [21]).

Due to the nature of the method used for the adjoint model, coding and in particular testing was time consuming requiring patience and care to avoid errors. This took more time than was expected and so delayed the project from advancing.

11 Conclusion

Many other tests were run apart from those results included in section §9 and in most cases a good estimate value for c_0 was obtained. However, the values for τ were often very far from the true value as was ∇J from the expected value of near zero. Even when ∇J was near zero, τ still did not approach it's true value.

We explain the result of τ varying and having marginal affect on the other values by saying the system is τ insensitive.

We can further explain the failure of ∇J to tend to zero in all cases by the convergence factor in CONMIN, that resulted in the process completing with good (convergent) values for c_0 and the cost function before minimisation of ∇J had been achieved. With a more powerful and accurate computer it is possible to increase the machine accuracy parameter and adjust the EPS value for CONMIN which would allow convergence to occur at a later stage in the processing, thereby allowing more opportunity for minimisation to occur.

The programs, although tested as well as could be in the time allowed, are still likely to contain errors which further testing could eradicate if time permitted. Also more tests could be made with different combinations and magnitudes for the parameters used by the model to improve the performance.

The model exhibited resilience to a wide range of guess values (for c_0^g and τ^g), generally achieving a good estimate for c_0. Good resilience was also recorded in the test for values of the covarience factor $\sigma \leq 1$.

It was also concluded that τ had insignificant influence on the minimisation of J and hence in estimating the state variable.

It had been hoped to use real data collected from the M25 motorway but this stage was not reached because of time restriction, but it is anticipated that this will be done following the completion of this dissertation.

12 Appendix A- Glossary

Anticipation: the changes made by drivers to changing traffic conditions around them. Used in Payne's equations.

Continuity equation: differential equation that describes the conservation of a conserved quantity.

Convection: changes in the mean traffic velocity caused be vehicles joining or leaving the flow. Used in Payne's equations.

Cost function: measures the bias between the observations and the model.

Data Assimilation: is the incorporation of observational data into a numerical model to produce a model state which most accurately describes the observed reality.

EPS: is the user supplied convergence parameter to CONMIN.

Hyperbolic: a system of partial differential equations is hyperbolic if for $\mathbf{x}_p + A(\mathbf{x})\mathbf{x}_q, A(\mathbf{x})$ is diagonalisable and has real eigenvalues.

Insensitive Parameter: ability of a model or system to be unaffected by widely ranging values of the parameter.

Relaxation: the tendency of traffic flow to approach an equilibrium velocity. Used in Payne's equations.

Stochastic process: a process whose behaviour is essentially non-deterministic, that is the system's subsequent state is a combination of the process's predictable actions and a random element.

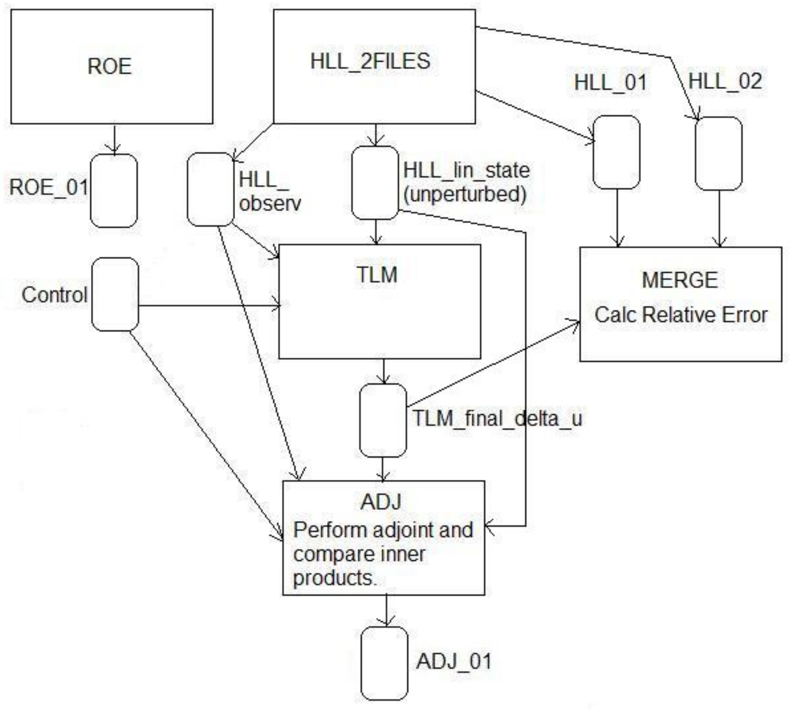
Tangent Linear approximation: An assumption used in applications of tangent linear models and adjoint models that the evolution of small perturbations in nonlinear models may be approximated by tangent linear (and adjoint) equations for finite time intervals (ref: 1).

TLM: Tangent Linear Model. A model, comprising tangent linear equations, that maps a perturbation vector, $\delta\mathbf{x}(t_1) = M\delta\mathbf{x}(t_0)$, from initial time t_0 to forecast time t_1. Where, M is the tangent linear operator and $\mathbf{x}$ is the model state vector.

Traffic flow model: formulates the relationships between traffic flow characteristics

13 Appendix B

13.1 Data Flow Diagram


ROE
HLL_2FILES
HLL_01
HLL_02
ROE_01
HLL_ observ
HLL_lin_state (unperturbed)
Control
TLM
MERGE Calc Relative Error
TLM_final_delta_u
ADJ Perform adjoint and compare inner products.
ADJ_01

13.2 Program Outlines

13.2.1 Introduction

All programs were written in Fortran using the Plato development application, a brief description of their function and the data input/output files is given below. A Control file is maintained of the initial values used by HLL together with other parameter values that are required to be passed between programs.

13.2.2 Roes

The first program of the unlinearised model from which the HLL program was developed.

13.2.3 HLL_2files

The Fortran implementation of the HLL model starts from initial values of the state vector and produces an output file of u at the final time step. The constants c_0 and τ are then randomly perturbed and the model re-run. The cost function is calculated for each condition.

Output:
Control file output file containing parameter values
HLL_lin_state output file of containing values of u at each x position for each time step
HLL_01 output file of final values of u using unperturbed values
HLL_02 output file of final values of u using perturbed values
HLL_observs output file of containing values of observations values at each x position for each time step

13.2.4 TLM

The program performs the TLM processing using the equivalent HLL values of u at the start of each 'x' step. Produces a file of the δu values for the final time step.

Input:
Control file containing parameter values (and perturbed state vector values)
HLL_lin_state containing initial all values of unlinearised u (to be used at the start of each 'x' step)
HLL_observs containing values of observations values at each x position for

each time step

Output:
TLM_01 containing final time step values of δu

13.2.5 ADJ

The program performs the Adjoint model processing, but in the reverse order to processing by HLL and TLM. (i.e. starting at the final time and working backwards to the start time, and within each time step beginning with the greatest x value and working backwards to the smallest.) Again the equivalent HLL values of u are used at the start of each $'x'$ step (HLL values being accessed in reverse order to which they were produced) together with the final time step values of δu from TLM for the first time step only.

The program includes the calculation of the inner products using the output of TLM and ADJ, to verify the correct working of the system.
Calculates inner product 1 = $(\delta c_0)^2 + (\delta \tau)^2 + (\delta u)^2$ (using the TLM_final_delta_u file values). Equivalent of $(M\delta x).(M\delta x)$.

Calculates inner product 2 = $\delta\tau.\delta\hat{\tau} + \delta c_0.\delta\hat{c_0}$ only (since the initial perturbations are zero).
Equivalent of $\delta x.M^T M\delta x$.
Where

- $\delta\tau$ is the perturbed value calculated and used by HLL, used for TLM and the initial value for ADJ. Similarly for δc_0.
- $\delta\hat{\tau}$ and $\delta\hat{c_0}$ are final values calculated by ADJ.

If the system is functioning correctly, then inner product 1 should equal inner product 2.
The results are displayed on the monitor.

The program includes the calculation of the gradient of the Cost function for use by CONMIN as detailed below:

$$\nabla J = 2 \cdot (\delta \hat{c}_0 + \delta \hat{\tau})$$

Input:
Control file containing parameter values
HLL_lin_state containing all values of unlinearised $\mathbf{u}$ (to be used at the start of each 'x' step)
TLM_final_delta_u containing final time step values of δu.
HLL_observs containing values of observations values at each x position for each time step

Output:
Control file updated to contain final values of $\delta \hat{c}_0$ and $\delta \hat{\tau}$
Results of inner products calculations.

13.2.6 Merge_err

Compares the values of δu produced from the difference in values between u on HLL_01 and HLL_02 and the δu values from TLM using a Relative Error calculation.

Input:
HLL_01 final values of u using unperturbed values
HLL_02 final values of u using perturbed values
TLM_01; TLM values of δu

Output to terminal the Relative errors of ρ and v.

13.2.7 CONMIN

CONMIN is a computer subroutine that finds the values of c_0 and τ that produce a minimisation of the cost function. In summary, it is called from a small Fortran program and then repeadily calls HLL and ADJ to calculate the J and ∇J respectively until convergence of the values is achieved.

13.3 Control file values

Record 1: N, c_0, τ, perturbation factor
Record 2: t, δt, δx
Record 3: perturbed value of c_0, perturbed value of τ (from HLL)
Record 4: observational error covariance for ρ, observational error covariance for v, Cost function value (all from HLL)
Record 5: $\delta\hat{c_0}$, $\delta\hat{\tau}$ (from ADJ)

14 Bibliography

References

[1] Danila Volpi, Estimation of Parameters in Traffic Flow Models Using Data Assimilation, 2009

[2] Department for Transport, vehicle licensing statistics, 2010, p.2

[3] Living Streets, http://www.livingstreets.org.uk/news/uk/-/driven-to-excess

[4] Chin Jian Leo, Robert L. Pretty, Numerical simulation of macroscopic continuum traffic models, 1990

[5] John C Strikwerda, Finite difference schemes and partial differential equations, second edition, 2004

[6] P L Roe, Approximate Riemann solvers, parameter vectors, and difference schemes, 1981

[7] A Harten, J M Hyman, and P D Lax, On finite difference approximations and entropy conditions for shocks, Comm. Pure and Appl. Math., 1976, 29:297322

[8] Allan R Robinson, Pierre F J Lermusiaux, Overview of data assimilation, Harvard University, 2000

[9] N K Nichols, Data assimilation for the Earth system, p.11

[10] E Kalnay, Atmospheric Modelling, Data assimilation and predictability, 2003, p.154

[11] A Lawless, S Gratton, N Nichols, An investigation of incremental $4D-$Var using non-tangent linear models, 2004, p.6-7

[12] D F Shanno, K H Phua, Remark on algorithm 500 - a variable method subroutine for unconstrained nonlinear minimization. *ACM Trans. on Mathematical Software,* 1980, p.6, 618622

[13] F Bouttier and P Courtier, Data assimilation Concepts and Methods, 1999

[14] N Nichols, Mathematical concepts of data assimilation, 2009

[15] Eugenia Kalnay, Ensemble prediction and strategies for initialization: Tangent linear and adjoint models, singular vectors, lyapunov vectors, 2008, Lecture 2

[16] Weiyu Yang, Michael Navon, Documentation of the tangent linear model and it's adjoint of the adiabatic version of the NASA GEOS-1 C-Grid GCM (Version 5.2), 1995, p.14-15

[17] Y Li, M Navon, W Yang, X Zou, J Bates, S Moorthi and R Higgins, Four-dimensional variational data assimilation experiments with a multilevel semi-Lagrangian semi-implicit general circulation model, Monthly weather review, May 1994, 966-983

[18] W C Chao, L-P Chang, Development of a four-dimensional variational analysis system using the adjoint method at GLA. Part 1: Dynamics. Monthly weather review, August 1992, 1661-1673

[19] A Lawless, N Nichols and S Ballard, A comparison of two methods for developing the linearisation of a shallow-water model, Quart. J. Royal Met Soc, 2003, 129:1237-1254

[20] I M Navon, X Zou, J Derber and J Sela, Variational data assimilation with an adiabatic version of the NMC spectral model, Monthly weather review, July 1992, 1433-1446

[21] Ronald M. Errico, What is an adjoint model, 1997

[22] J F Müller, T Stavrakou, Inverse Modelling of CO Emissions, Belgian Institute for Space Aeronomy, 2006