UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

USING LaTeX TO WRITE A DISSERTATION AT O.U.

A THESIS

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirements for the

Degree of

MASTER OF SCIENCE IN METEOROLOGY

By

BRIAN FIEDLER
Norman, Oklahoma
1998

USING LaTeX TO WRITE
A DISSERTATION AT O.U.

A THESIS APPROVED FOR THE
SCHOOL OF METEOROLOGY

By

Prof. Albert Einstein

Prof. Enrico Fermi

Prof. Alfred E. Newman

Dedication

This dissertation is dedicated to Donald Knuth and Linus Torvalds.

A dedication page is optional in the OU dissertation. This section is not allowed to have a printed pagenumber, nor is it allowed to increment the page number counter. This is why we use the hacked `\prefacesectionX` for the Dedication, rather `\prefacesection`.

Acknowledgements

I would like to thank Prof. Einstein and Prof. Fermi for their expert advice and patience. Their sacrifice was enormous, considering the fact that I got a D in freshman physics — and I never really got much better after that.

Contents

List Of Tables

List Of Figures

Abstract

In 1998, I hacked the `OUdisseration.cls` file from the `USCthesis.cls` file that I grabbed from the web. As of April 21, 2009, this "dissertation" uses the `OUdisseration5.cls` to define the document class. The comments inside `OUdissertation5.cls` describe the changes. (You may see a bug here with the text extending past the margin.)

I put this example "dissertation" together for the benefit of the students at the University of Oklahoma, who can use it as a template for their own work. Depending on the demand of the users, maybe someday I will clean this up. *Caveat user.*

Chapter 1

Introduction

Daley (1991) and Ghil and Malanotte-Rizzoli (1991) refer to this problem as the "data assimilation" problem. But I can write this sentence a different way: Consequently, this problem is referred to as the "data assimilation" problem (Daley 1991; Ghil and Malanotte-Rizzoli 1991). I will let the text run on here a bit so that a big reference list will be generated.

Formally, let c be a control vector of size m and S denote the feasible region for the control vector. For any c in S, let $J(c)$ denote the weighted sum of the squared difference between the observation and the solution of the model corresponding to the control vector c. Except in trivial cases the explicit form of J as a function of c is not known. It is to be emphasized that there are other possible choices for the J function. One may be interested only in the state of the model at a given time instant, say $t = \Delta$. Whatever be the nature and type of the J function, mathematically, the data assimilation problem can be stated as follows : find a c^* in S such that $J(c^*)$ is a minimum, that is, we are lead to an optimization problem under the dynamical constraints of the model equations. Since J is a "smooth" function, one method for finding c^* is to use one of the many variants of the classical gradient method. This is however, more easily said than done. The difficulty primarily stems from the fact that J is not known explicitly. A now popular method for finding the gradient of J is called the "adjoint" method (Ghil and Malanotte-Rizzoli 1991; Thacker and Long 1988). A summary of data assimilation using the adjoint method is shown in Fig. 3.1 and described below:

Considerable success has been reported in the literature in the use of adjoint method for finding c^* (Luenberger 1973; Sun and Flicker 1991; Wolfsberg 1987). The success of this combination is largely dependent on the properties of the J function. This approach can succeed only if J is unimodal in S. It turns out that the modality of J critically depends on the model dynamics. It is now known that the nonlinearity in the model dynamics induces multimodality in the J function (Li et al. 1991; Chung 1996).

There are at least two factors affecting the rate of convergence of the iterates leading to the optimum value of the J function. First, is the number and distribution of the observations. There is a minimum number of observations required from an algorithmic viewpoint, but satisfaction of this requirement is insufficient to guarantee a solution. The distribution of these data (in space and time), in concert wtih the dynamics, dictates the existence of a solution. Second, is the shape of the J function. Judicious choice of scaling can remove eccentricities in the J-field. It is thus imperative to understand the role of these two factors affecting the iterates.

The effect of the number and distribution of observations on the quality of the iterates are often examined using controlled experiments which have come to be known as the "twin" experiments. In this, a point in the feasible region for the control vector is first chosen and then the model solution is calculated for this value of the control vector . Then observations (including known error) are generated from the model solution by adding noise with known characteristics. By computing the optimal estimate of the control vector for different sets of observations, we can develop a better understanding of the dependence of the optimal estimate on the number, distribution and accuracy of observations.

As for the shape of the J function, since it is not known explicitly, we must be contented with the analysis of the properties of J around the local minima. This is often done by approximating J around the optimum c^* using a quadratic form such as

$$J(c) = \frac{1}{2}c^t Hc + p^t c + q, \tag{1.1}$$

where H is the Hessian matrix which is a symmetric matrix of the second derivatives of J with respect to c,

$$p = (p_1, p_2, p_3, p_4, ..., p_n)^t \tag{1.2}$$

is a vector and q is a constant. By analyzing the eigenvalues of H, we can draw inferences on the shape of J in the vicinity of c^* (Thacker 1989). For example, if one of the eigenvalues of H is very small, then the contours or the level curves of J for the valley around c^* are elongated ellipses. This would imply that the iterative process of locating the minimum would converge slowly and with difficulty.

In this dissertation, one aim is to apply the adjoint method to the mixed layer model (Ball 1960; Lilly 1968). This model is used to predict the return flow of the warm, humid air from the Gulf of Mexico into the coastal plains during the winter months. The mixed layer model has a small number of unknown variables/parameters. Athough small, the model is nonlinear and thus presents difficulties in dynamical optimization. With the experience gained, we wanted to analyze the computational aspects of data assimilation. This brought us to the shallow water equations (Pedlosky 1987), which has a larger number of variables in the discrete model. We are interested in solving some of the computational aspects of this model that are relevant to data assimilation. Since the solution of the forward model and the adjoint take a good fraction of the efforts, we turned our attention to parallel methods for solving this class of equations.

The shallow water model is discretized first using the Euler scheme. But due to its instability, we also examined the solution using a stable leapfrog scheme. Both types of discretization resulted in a block bi-diagonal system of equations for the forward model and the adjoint model. We are interested in examining the comparative analysis of four vector/parallel algorithms in solving this system of equations. These four vector/parallel algorithms belong to a class of direct parallel methods. The first algorithm is also known as the "divide and conquer" method (Lakshmivarahan and Dhall 1990). The second algorithm is a variation of the divide and conquer method (Lakshmivarahan and Dhall 1990; Conn and Podrazik 1994; Meyer and Podrazik 1987; Van der Vorst 1988; Meyer and Podrazik 1989). The third algorithm is known as the partition algorithm (Van der Vorst and Dekker 1989). The fourth algorithm is the cyclic reduction method (Lakshmivarahan and Dhall 1990). A comparison of these four vector/parallel algorithms is done on the CrayJ90 in scalar mode and using 1,2,4 and 8 vector processors.

This dissertation is organized as follows. A description of the use of encapsulated postscript files is presented in Chapter 2. One way to present bargraphs is shown in model equations are described in Section 3.1. Some example tables are shown in Section 3.2.

Chapter 2

Mostly `.eps` and `.pdf` examples

This chapter shows how encapsulated postscript figures can be included in the document. In particular, we wish to show how raw `.eps` and `.pdf` files can annotated and labeled to make them of "publication quality".

2.1 A few easy equations and a picture

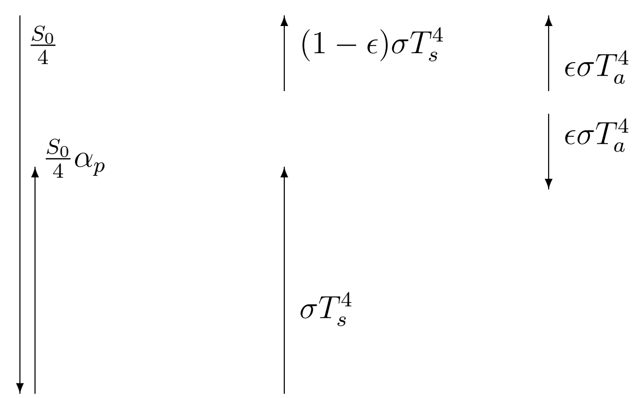

Figure 2.1: Radiative equilibrium with a "greenhouse effect". This picture was drawn using LaTeX `epic` commands in the so-called *picture environment*. With `pdflatex`, the horizontal lines will be missing. `pdfprob.tar.gz` shows the more complicated procedure to include epic figures with `pdflatex`.

A radiative balance at the surface requires that:

$$\frac{S_0}{4}(1-\alpha_p) + \epsilon\sigma T_A^4 = \sigma T_s^4. \tag{2.1}$$

Radiative equilibrium of the putative "atmosphere" in Figure 2.1 requires that:

$$\epsilon\sigma T_s^4 = 2\epsilon\sigma T_s^4. \tag{2.2}$$

We use (2.2) to eliminate T_A from (2.1), which gives:

$$\frac{S_0}{4}(1-\alpha_p) + \frac{1}{2}\epsilon\sigma T_s^4 = \sigma T_s^4. \tag{2.3}$$

or

$$T_s = \sqrt[4]{\frac{S_0(1-\alpha_p)}{4\sigma(1-\frac{\epsilon}{2})}} \tag{2.4}$$

2.2 Simple inclusion of `.eps` or `.pdf` graphics

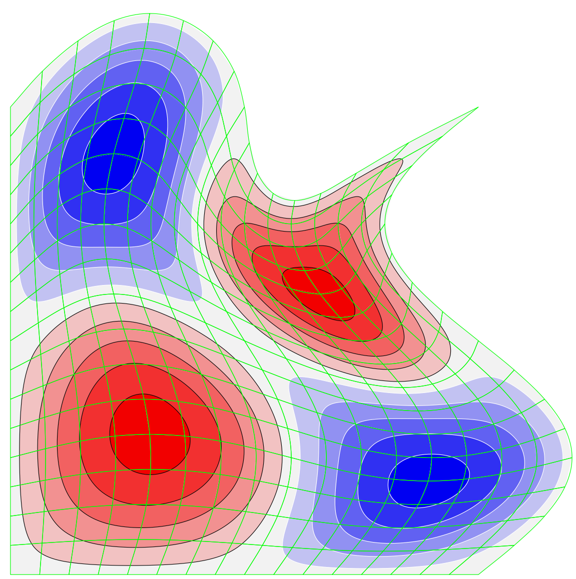

Figure 2.2: This is what happens when you leave your figure in a hot car during July.

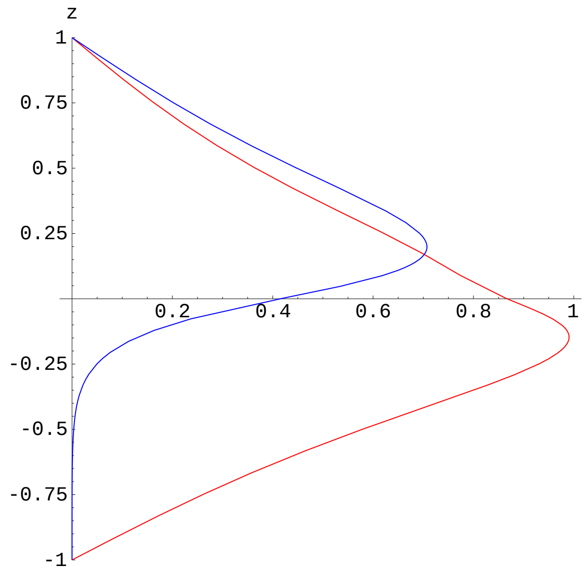

Figure 2.3: Raw `.eps` from **Mathematica**.

2.3 `.eps` or `.pdf`, annotated

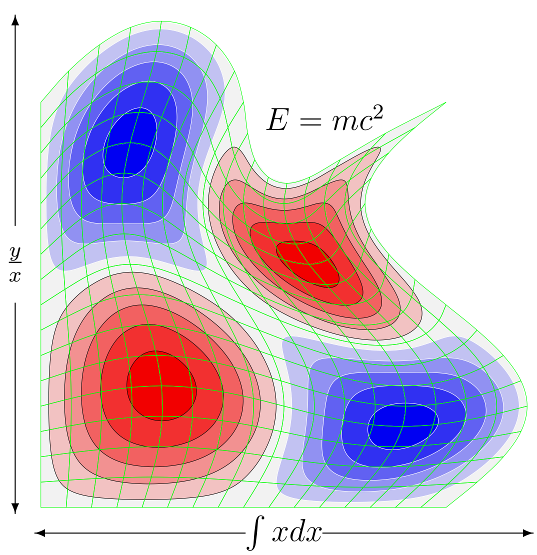

Figure 2.4: The melted figure with annotations. Wonderful! All of the epic commands here work in both `latex` and `pdflatex`, in Fig. 2.1 some did did not in `latex`. This is may favorite way to put labels on my figures.

2.4 Using the `psfrag` system with .eps

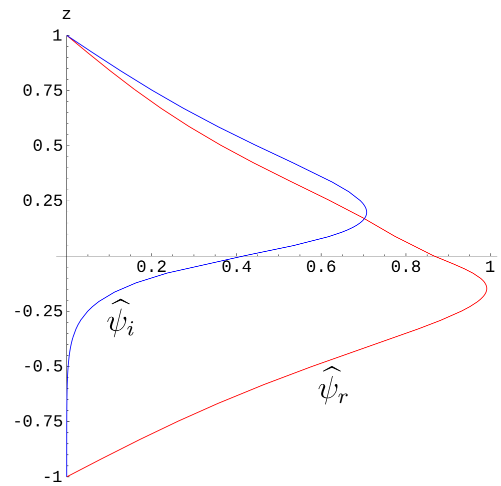

Figure 2.5: This looks ugly until rendered in postscript. At that time, the `z` (which has now vanished) is turned into a ζ. This will not work with `pdflatex`.

Chapter 3

Other examples

Note that LaTeX will place Figure 3.1 after the list, because I did not use the `[H]` option after the `\begin{figure}`.

Step 1: Make a first estimate of the initial condition (I.C.).

Step 2: Run the forward model to generate a forecast over the assimilation interval.

Step 3: Compute the cost function, J, using the observations and the model output.

Step 4: Compute the gradient of the cost function with respect to the control variable, $\nabla J(c)$, by integrating the adjoint model (which is a combination of the forward model and cost functional) backwards using the Lagrangian multiplier method.

Step 5: Generate a new estimate of the initial condition (I.C.) for another forecast using a minimization procedure (*e.g.*, Steepest Descent or Conjugate Gradient Method). This is done iteratively toward the minimum of the cost function using the calculated cost function and its gradient.

Step 6: The process is repeated until some convergence criteria are met, *i.e.*, the cost functional, J, is near its local minimum. In practice, this is determined by an insignificant change in the gradient from one iteration to the next. If not, the process will be terminated when some maximum number of iterations have been achieved.

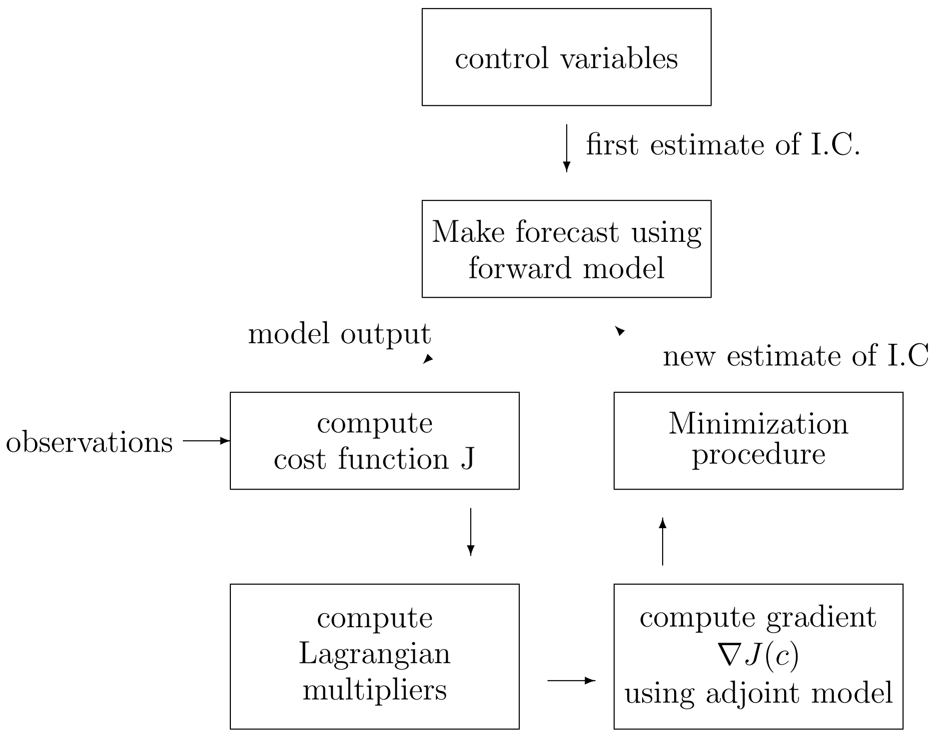

Figure 3.1: Flowchart of data assimilation using the adjoint method.

Best dense, row-wise algorithms

Algorithm 1, p=2, gaxpy

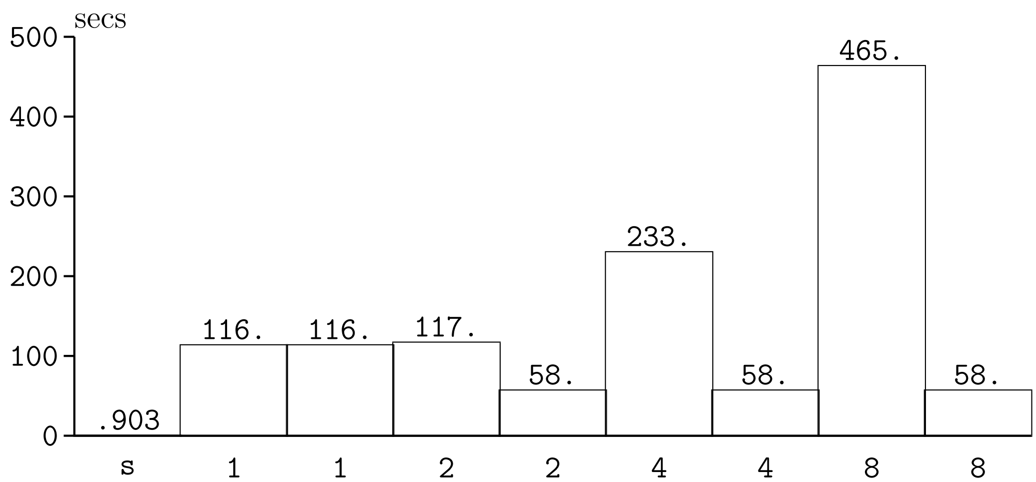

Figure 3.2: CPU time (hatched) and wall clock time (white) for srowdg (s) and a1rowdg with various numbers of processors. m=128 N=64

3.1 A Bargraph

Hey, where did that bargraph go? I thought I put it in this section!

3.2 Some tables

I will put some text here just to see what happens to it.

time (hours)	T (C)
0	17.0
1.5	18.8
3.0	19.8
4.5	20.5
6.0	21.0
7.5	21.3
9.0	21.5
10.5	21.8
12.0	22.2
13.5	22.8
15.0	23.5
16.5	24.1
18.0	24.3

Table 3.1: T(t) for the initial 18 hours

Step 1	Step 2		
	Stage 1	Stage 2	Stage 3
x_1			
$x_{1:2}$			
$x_{1:3}$			
$x_{1:4}$			
x_5	$x_{1:5}$		
$x_{5:6}$, $A_{5:6}$	$x_{1:6}$		
$x_{5:7}$, $A_{5:7}$	$x_{1:7}$		
$x_{5:8}$, $A_{5:8}$	$x_{1:8}$		
x_9		$x_{1:9}$	
$x_{9:10}$, $A_{9:10}$		$x_{1:10}$	
$x_{9:11}$, $A_{9:11}$		$x_{1:11}$	
$x_{9:12}$, $A_{9:12}$		$x_{1:12}$	
x_{13}			$x_{1:13}$
$x_{13:14}$, $A_{13:14}$			$x_{1:14}$
$x_{13:15}$, $A_{13:15}$			$x_{1:15}$
$x_{13:16}$, $A_{13:16}$			$x_{1:16}$

Table 3.2: Algorithm # 3 example

3.3 Some complicated equations

To this end, the Lagrangian is introduced (see Lanczos (1970) for the theoretical foundations of minimizing functions subject to a constraint).

$$
\begin{aligned}
L(c, \lambda, \mu, \eta) &= J(c) \\
&\quad + \sum_{i=1}^{n} \lambda_i \left[\theta_i - \theta_{i-1} - \frac{\tau}{h_{i-1}} C_T V (1+K) \left(T_{i-1} - \theta_{i-1} \right) \right] \\
&\quad + \sum_{i=1}^{n} \mu_i \left[h_i - h_{i-1} - \tau K C_T V \frac{\left(T_{i-1} - \theta_{i-1} \right)}{\sigma_{i-1}} - \tau W \right] \\
&\quad + \sum_{i=1}^{n} \eta_i \left[\sigma_i - \sigma_{i-1} + \frac{\tau}{h_{i-1}} C_T V (1+K) \left(T_{i-1} - \theta_{i-1} \right) \right. \\
&\qquad \left. - \tau \gamma K C_T V \frac{\left(T_{i-1} - \theta_{i-1} \right)}{\sigma_{i-1}} - \tau \gamma W \right] \qquad (3.1)
\end{aligned}
$$

where $\lambda = (\lambda_1, \ldots, \lambda_n)^t$, $\mu = (\mu_1, \ldots, \mu_n)^t$, $\eta = (\eta_1, \ldots, \eta_n)^t$ are the undetermined Lagrangian multipliers in (3.1).

$$
E = \begin{bmatrix}
-I_3 & 0 & 0 & 0 & \ldots & 0 & 0 & 0 \\
D_1 & -I_3 & 0 & 0 & \ldots & 0 & 0 & 0 \\
0 & D_2 & -I_3 & 0 & \ldots & 0 & 0 & 0 \\
\ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\
0 & \ldots & \ldots & \ldots & \ldots & \ldots & D_{n-1} & -I_3
\end{bmatrix}, \qquad (3.2)
$$

By definition,

$$
\begin{aligned}
\epsilon_i^{\theta} &= \theta_i - \overline{\theta}_i, & (3.3) \\
\epsilon_i^{h} &= h_i - \overline{h}_i, & (3.4) \\
\epsilon_i^{\sigma} &= \sigma_i - \overline{\sigma}_i. & (3.5)
\end{aligned}
$$

Now we can still refer to (3.2) or (3.3)-(3.5).

Bibliography

Ball, F. K., 1960: Control of inversion height by surface heating. *Quart. J. Roy. Meteor. Soc.*, **86**, 483–494.

Chung, W. L., 1996: *Analysis of data distribution in variational assimilation schemes of interest in meteorology using Burger's equation*. Ph. d. dissertation, Department of Computer Science, Univ. of Oklahoma.

Conn, H. E. and L. J. Podrazik, 1994: Parallel recurrence solvers for vector and SIMD supercomputers. *J. of Parallel and Distributed Computing*, **23**, 435–445.

Daley, R., 1991: *Atmospheric Data Analysis*. Cambridge Uni. Press.

Ghil, M. and P. Malanotte-Rizzoli, 1991: Data assimilation in meteorology and oceanography. *Advances in Geophysics*, **33**, 141–265.

Lakshmivarahan, S. and S. Dhall, 1990: *Analysis and Design of Parallel Algorithms: Arithmetic and Matrix problems*. McGraw-Hill.

Lanczos, C., 1970: *The Variational Principles of Mechanics*, Univ. of Toronto Press, chapter 2.

Li, Y., K. K. Droegemeier, and J. M. Lewis, 1991: Multiple minima in the cost functional of variational four-dimensional data assimilation methods : Their origin and role in the predictability of nonlinear dynamical systems. *9th conference on Numerical Weather Prediction*.

Lilly, D. K., 1968: Models of cloud-topped mixed layers under a strong inversion. *Quart. J. Roy. Meteor. Soc.*, **94**, 292–309.

Luenberger, D. G., 1973: *Introduction to Linear and Nonlinear Programming*. Addison Wesley.

Meyer, G. and L. Podrazik, 1987: A parallel first-order linear recurrence solver. *J. of Parallel and Distributed Computing*, 117–132.

—, 1989: Parallel iterative algorithms for optimal control. *Proceedings of the Third SIAM Conference on parallel processing for scientific computing*, 250–254.

Pedlosky, J., 1987: *Geophysical Fluid Dynamics*. Springer-Verlag.

Sun, J. and D. W. Flicker, 1991: Recovery of three-dimensional wind and temperature fields from simulated single-doppler radar data. *J. Atmos. Sci.*, **48**, 876–890.

Thacker, W. C., 1989: The role of the Hessian matrix in fitting models to measurements. *Journal of Geophysical Research*, **94**, 6177–6196.

Thacker, W. C. and R. B. Long, 1988: Fitting dynamics to data. *Journal of Geophysical Research*, **93**, 1127–1240.

Van der Vorst, H. A., 1988: Parallel solution of bidiagonal systems coming from discretized pde's. *IEEE Trans. Magnetics*, **24**, 286–290.

Van der Vorst, H. A. and K. Dekker, 1989: Vectorization of linear recurrence relations. *SIAM J. Sci. Sta. Comput.*, **10**, 27–35.

Wolfsberg, D. G., 1987: Retrieval of three-dimensional wind and temperature fields from single doppler radar data. Report, CIMMS, U. of Oklahoma.

Appendix A

A Long Proof

The buoyancy production term requires a closure for $\overline{w'\theta_v'}$. In saturated conditions $q_l = q_s$ and we have 3 linear relations between 5 flux densities:

$$\begin{aligned}
\overline{w'\theta'} &= \overline{w'\theta_l'} + B\overline{w'q_t'} - B\overline{w'q_s'} && \text{(A.1)}\\
\overline{w'\theta_v'} &= C\overline{w'\theta'} + D\overline{w'q_s'} - E\overline{w'q_t'} && \text{(A.2)}\\
\overline{w'q_s'} &= F\overline{w'\theta'} && \text{(A.3)}
\end{aligned}$$

where

$$\begin{aligned}
B &\equiv \chi\frac{L}{c_p} && \text{(A.4)}\\
C &\equiv 1 + 1.61\overline{q_s} - \overline{q_t} && \text{(A.5)}\\
D &\equiv 1.61\overline{\theta} && \text{(A.6)}\\
E &\equiv -\overline{\theta} && \text{(A.7)}\\
F &\equiv 0.622\frac{L\overline{q_s}}{R_d\overline{T}\overline{\theta}} && \text{(A.8)}
\end{aligned}$$

We will eliminate $\overline{w'q_s'}$ and $\overline{w'\theta'}$ from (A.1)-(A.3) and thus derive a linear relation between $\overline{w'\theta_v'}$ and $\overline{w'\theta_l'}$ and $\overline{w'q_t'}$:

$$\overline{w'\theta_v'} = \frac{C+DF}{1+BF}\overline{w'\theta'} + \left(\frac{C+DF}{1+BF}B - E\right)\overline{w'q_t'} \tag{A.9}$$

Eq. (A.3) has been derived from the Clausius-Clapyron equation

$$\frac{de_s}{dT} = e_s\frac{L}{R_dT^2} \tag{A.10}$$

and

$$q_s \equiv 0.622\frac{e_s}{p} \tag{A.11}$$

followed by assumptions that $\frac{p'}{\overline{p}}$ is small compared with both $\frac{1}{e_s}\frac{de_s}{dT}T'$ and $\frac{T'}{\overline{T}}$. Here is how we calculate λ:

$$e(z) = \int_z^{z+\lambda_{up}(z)} \frac{g}{\overline{\theta_v}(z')} \left[\overline{\theta_v}(z) - \overline{\theta_v^*}(z')\right] dz' \tag{A.12}$$

$$e(z) = \int_z^{z-\lambda_{down}(z)} \frac{g}{\overline{\theta_v}(z')} \left[\overline{\theta_v}(z) - \overline{\theta_v^*}(z')\right] dz' \tag{A.13}$$

$$\lambda(z) = \left[\lambda_{down}(z)\lambda_{up}(z)\right]^{1/2} \tag{A.14}$$