Neural Networks for Atmospheric Retrievals

Howard E. Motteler
NRC, Code 930
NASA/GSFC
Greenbelt, MD 20771

J. A. Gualtieri
USRA, Code 902.2
NASA/GSFC
Greenbelt, MD 20771

L. Larrabee Strow
Department of Physics
UMBC
Baltimore, MD 21228

Larry McMillin
NOAA/NESDIS
5200 Auth Road
Camp Springs, MD 20746

Abstract

We use neural networks to perform retrievals of temperature and water fractions from simulated clear air radiances for the Atmospheric Infared Sounder (AIRS). Neural networks allow us to make effective use of the large AIRS channel set, and give good performance with noisy input. We retrieve surface temperature, air temperature at 64 distinct pressure levels, and water fractions at 50 distinct pressure levels. Using 728 temperature and surface sensitive channels, the RMS error for temperature retrievals with 0.2K input noise is 1.2K. Using 586 water and temperature sensitive channels, the mean error with 0.2K input noise is 16%. Our implementation of backpropagation training for neural networks on the 16,000-processor MasPar MP-1 runs at a rate of 90 million weight updates per second, and allows us to train large networks in a reasonable amount of time. Once trained, the network can be used to perform retrievals quickly on a workstation of moderate power.

1 Introduction

The next generation of NASA earth viewing satellites on Earth Observing System (EOS) platforms will produce a deluge of raw data that must be processed into products that describe the state of the earth and its atmosphere over time. Satellite instruments that probe the atmosphere measure radiances over a number of channels, and this information must be "inverted" to obtain information about the atmospheric state, such as the temperature, humidity, and composition.

The Atmospheric Infrared Sounder (AIRS) [3], currently under development, should provide both higher accuracy and vertical resolution than the present operational sounders (HIRS/MSU) [10], and lead to higher forecasting skill and a long term accurate measure of climate change. The AIRS instrument will contain upwards of 4000 channels at a much higher spectral resolution than the currently operational HIRS instrument, which

has 20 channels. The optimum use of these data for atmospheric sounding in a cost effective way may require completely new techniques, as existing methods for current instruments may not be transferable in a straightforward manner. Traditional retrieval (or inversion) techniques are computationally intensive, especially non-linear techniques that require several iterative calculations of the channel radiances. It is estimated that the AIRS will require one of the most computationally intensive data systems on EOS.

To address these new computational challenges, we have implemented a backpropagation training algorithm on the Maspar MP-1 at Goddard Space Flight Center to train neural networks to perform atmospheric retrievals of temperature and water profiles from simulated clear air radiances for the AIRS instrument. [The problem of cloudy atmospheres is a topic of future work not treated here.] These neural networks allow us to make effective use of the large AIRS channel set, give good performance with noisy input data, and allow for very fast processing even with very large numbers of channels.

We have found that the backpropagation code maps very well to the Maspar, and we have obtained network training rates of 93 million connection updates per second (CUPS) in single precision [1]. Once such a network has been trained on the Maspar, it can be downloaded to a workstation where the time to obtain retrievals is the time to perform three matrix multiplies – of order less than 0.5 sec with a thousand input channels. (On the Maspar the retrieval time is at least an order of magnitude faster).

The accuracy of the results obtained with our neural networks are quite competitive with other retrieval methods. Using 728 temperature and surface sensitive channels, and with 0.2K std noise added to the input brightness temperatures, the neural network has an overall RMS error retrieving 64 pressure levels of 1.22K. Using 586 water, surface, and temperature sensitive channels, and with 0.2K std noise added, the neural network has an overall error retrieving 50 pressure levels of 16% [2].

In order to better understand retrieval performance, we perform a sensitivity analysis of trained networks. This analysis is useful in selecting what sets of channels are to be used, in a process of iterative refinement, and in many cases shows a close correspondence to plots of weighting functions (discussed in the next section).

In the sequel we describe the atmospheric retrieval problem, show how we use neural networks to solve the problem, describe the datasets used in training the networks, and present a number of representative results. We also describe the method of sensitivity analysis for evaluating the effectiveness of input sets to a neural network.

2 Atmospheric Retrievals

The problem of atmospheric retrievals [7], [5] (the "inverse problem") is to take as input the radiances at a specified set of frequency channels measured by a sensor on a satellite above the top of the atmosphere and compute the temperature or water profiles of the atmosphere (as a function of pressure) that gave rise to those radiances.

Associated with the inverse problem is the "forward problem" of computing the radiances

at the top of the atmosphere generated by layers of molecules in local thermal equilibrium from the surface up through the atmospheric column in the sensor's field of view. (We refer to this column as a temperature profile.) Assuming a plane parallel atmosphere in local thermodynamic equilibrium and negligible scattering, and no instrument function one can write the monochromatic radiance at nadir at the top of the atmosphere as

$$R_\nu = \epsilon_\nu B_\nu(T_s)\tau_\nu(P_s,[T(P)]) + \int_{\ln P_s}^{\ln \bar{P}} d\ln P \frac{d\tau_\nu(P,[T(P')])}{d\ln P} B_\nu[T(P)]$$

where ϵ_ν is the emissivity of the surface s, and the contribution of reflected radiation which is negligible at most frequencies of interest has not been included. $B_\nu(T)$ is the Planck function for emitted radiance of a blackbody at frequency ν and temperature T,

$$B_\nu(T) = 1.19\,x\,10^{-5}\frac{\nu^3}{\exp[1.439\nu/T]-1}.$$

The quantity $\tau_\nu(P_s,[T(P')])$ is the atmospheric transmittance from the surface at pressure P_s to the top of the atmosphere at pressure $\bar{P}$ which is the fraction of photons of frequency ν emitted at the surface P_s that arrive at the sensor at altitude $\bar{P}$. The quantity $\frac{d\tau_\nu(P,[T(P')])}{d\ln P}$ is the *weighting function* for the frequency ν and when multiplied by $d\ln P$ describes the fraction of photons of frequency ν emitted in the layer between pressure P and $P+dP$ that reach the top of the atmosphere. Fig. 1 [3] shows a few of the several thousand weighting functions available from the AIRS instrument and indicates how a weighting function can be associated with a narrow vertical region of the atmosphere. The notation $(P,[T(P')])$ as the argument of $\frac{d\tau_\nu}{d\ln P}$ is used to stress that it is *functional* of the profile $T(P')$ between $\bar{P}$ and P and a *function* of P.

Present retrieval systems are most easily classified as being either linear regression techniques or non–linear iterative techniques. Both techniques can use varying amounts of statistics for regularizing their solutions, as well as varying amounts of the forward problem radiative transfer. The linear regression approach is dependent on a very good first guess in order to be in the linear regime for the regression. The non-linear iterative method does not require such a good first guess, but does require time-consuming forward problem calculations. In addition, it is not clear if the non-linear iterative approach can coherently use all the information in the AIRS channel radiances without numerical problems. It may also be possible to iterate the linear regression approach, however this would result in the need to iteratively calculate the forward problem for a very large number of channels, introducing a very heavy computational burden.

3 Neural Networks

We use a three-layer feed-forward neural network, batch trained with a modified backpropagation algorithm [6], [8] with an adaptive learning rate. This network can be represented as

$$\mathbf{Y} = F_3(\mathbf{W}_3F_2(\mathbf{W}_2F_1(\mathbf{W}_1\mathbf{X}+\mathbf{B}_1)+\mathbf{B}_2)+\mathbf{B}_3),$$

where each F_i maps matrices to matrices, element by element, by applying a *transfer func-*

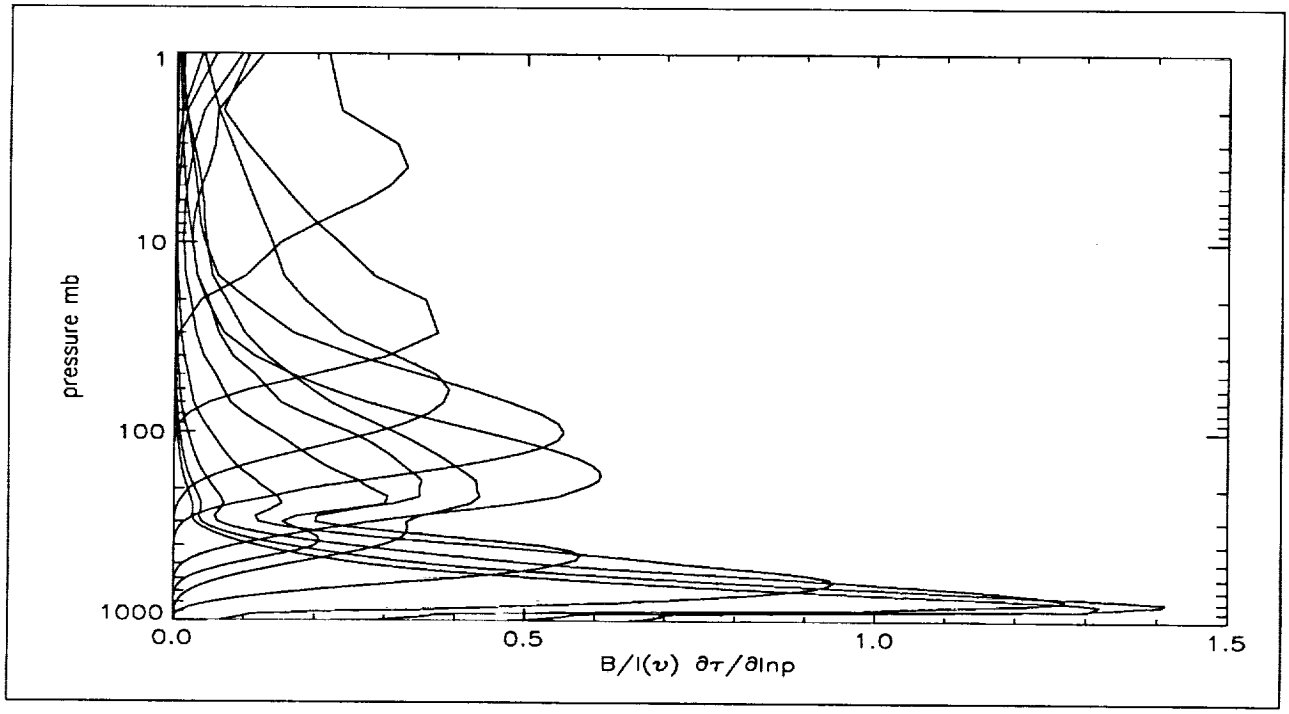

Figure 1: Representative weighting functions for the AIRS instrument. The x axis is a measure of the weighting function (where I(ν) is the radiance) and the y axis is pressure in mb.

tion to each matrix element and the matrices shown in boldface type are combined by matrix multiplication and addition. The mapping F_i is often referred to as a *layer*, with the weight matrices representing connections between layers. We use the hyperbolic tangent as a transfer function in the first two layers, and a linear function in the third. The input matrix $\mathbf{X}$ is of size (row $\times$ col) $n_{in} \times n_{training}$ and the output $\mathbf{Y}$ matrix is of size $n_{out} \times n_{training}$. The $\mathbf{W}_i$ are *weight matrices* of size respectively $n_1 \times n_{in}$, $n_2 \times n_1$, and $n_{out} \times n_2$. The $\mathbf{B}_i$ are *bias matrices* of respective sizes $n_1 \times n_{training}$, $n_2 \times n_{training}$, and $n_{out} \times n_{training}$ composed of single bias column vectors of respectively size n_1, n_2, and n_{out} replicated $n_{training}$ times to build the bias matrices. The quantities $n_{in}, n_1, n_2, n_{out}$, and $n_{training}$ are the number of input units (frequency channels), the number of first layer hidden units, the number of second layer hidden units, the number of output units (pressure levels), and the number of examples in the training set.

The networks we use for temperature retrievals have one input component for each instrument channel, and one output component for each AIRS pressure level. The first layer has between 90 and 108 transfer functions, the second between 60 and 72 transfer functions, and the output layer has a linear function for each pressure level. For water retrievals we have used 90 transfer functions in the first layer and 60 in the second layer.

Back-propagation training is a variation of gradient descent, in which weight and bias vectors are incrementally adjusted in an attempt to match the network output with a

set of training examples. This training set is a set of pairs, where each pair is an input together with the desired output. A single presentation of all the training data and corresponding weight and bias adjustment is called an *epoch*. Training consists of a sequence of epochs, and typically continues until the sum-squared error is acceptable or some resource limit is encountered. Training is a computationally intensive process for non-trivial networks. Although training is slow, applying a trained net is very fast, with the runtime being dominated by the time for the three matrix-vector multiplies.

It is convenient in the case of temperature retrievals to convert radiances R_ν to brightness temperatures Θ_ν according to the relation $B_\nu(\Theta_\nu) = R_\nu$ [9]. The brightness temperature is the temperature a blackbody would be at to produce the radiance R_ν. By doing this the large dynamic range of radiances is reduced to a much smaller dynamic range of brightness temperature. Further, each element of the input and output vector pairs are scaled to be differences from the mean values over the training set, and are divided by the standard deviation of the training set. This "normalizes" the inputs and outputs to a useful dynamic range for the transfer functions used.

We have developed a backpropagation code for the 128 $\times$ 128 processor MasPar MP-1 at the Goddard Space Flight Center in mpl (Maspar's parallel extension of C), which makes extensive use of the Maspar linear algebra library. This code efficiently handles the virtualization needed to map very large networks of many tens of thousands of weights and biases across the 16384 processing elements of the machine. Originally the code was written completely in double precision (64 bits) but since the results were found to be highly immune to noise in the data sets, a single precision version is now being used. Profiling tests show the code spends 95% of the time performing matrix multiplications, for which the Maspar routines are highly optimized. We are observing execution rates of 93 million weight updates a second [1] on typical datasets.

4 Datasets for Training

Datasets for training and testing are generated from the set of 1761 TIGR profiles [4] of temperature and water using the radiative transfer equation, to obtain corresponding radiances for the entire AIRS channel set. Thus the physics of the problem is built in by (1) the judicious selection of a large representative set of profiles and (2) the radiative transfer equation that gives the matching radiances. The TIGR profiles have been interpolated from the original 40 levels to either 66 TOVS pressure levels (for earlier experiments) or 64 TOVS pressure levels (as used in the AIRS science teams "write test"). The retrieved quantities are the temperatures and water amounts in the 64 intervening slabs with an additional element for the surface temperature, which may be different from the lowest slab. The surface emissivity is assumed to be one, for these experiments.

Our general method is to partition a dataset into training and extrapolation sets. The net is trained on the training set, and is then tested with the extrapolation set, both with and without noise; the noise inputs have a normal distribution and 0.2K standard deviation.

5 Results

In this section we present representative results for several profile and channel sets. In general, training runs were stopped when the RMS training error stopped showing significant improvement; this occurred after on the order of 100,000 epochs. Once network parameters (adaptive learning parameters, sizes of hidden layers, and initial distributions) are fixed in a useful range, different sets of random initial weights typically have a small effect on final RMS error. When the full set of TIGR profiles is divided into training and extrapolation sets of approximately equal size (with representatives from all latitudes in both sets) exchanging training and extrapolation subsets also has a small effect. The result for all the runs discussed are summarized in Table 1.

In run 150, the 880 even numbered TIGR profiles were used for training and the 881 odd numbered TIGR profiles were used for testing the network. Input to the net is brightness temperature for 666 AIRS channels, selected for surface and air temperature sensitivity. Output is surface temperature and air temperature at 66 distinct pressure levels. The network has 108 hyperbolic tangent transfer functions in the first hidden layer, and 72 hyperbolic tangent transfer functions in the second hidden layer. After 140,000 epochs, RMS training error is 1.20K, RMS extrapolation (testing) error is 1.26K, and RMS extrapolation error with 0.2K std noise is 1.44K. These results are shown in Fig. 2. After 100,000 epochs of further training with noisy data (0.2K std noise added to the input data), RMS training error is 1.22K, RMS extrapolation error is 1.23K, and RMS extrapolation error with 0.2K std noise is 1.37K

In the upper plot of Fig. 2, the temperature retrieval error at the surface and at each of 66 pressure levels is shown. In the lower plot, the same set of errors is presented as 11 groups of 6 pressure levels (the surface is still distinct, and is not grouped with any pressures levels). We do not have a completely satisfactory explanation for the small 'oscillations' in the 66 level plot. This pattern of fine variations appears across a wide range of training sessions and channel sets. (Note the similarity between these small scale variations in the Fig. 2 and Fig. 3 plots.) One possible explanation is that these variations correspond to variations in the numbers of weighting functions available at different pressure levels. Another possibility is that these may be an artifact of the fast transmittance code (as supplied by JPL for the AIRS science teams "write test") that we use to generate brightness temperatures. This is a matter for further investigation.

A sensitivity analysis of run 150 (discussed in the next section) is shown in Fig. 4. This analysis, together with similar results from other runs using the same channel set, indicated that channels with wavenumbers roughly between 750 and 1200 were not being used by the network. This information, together with the relatively high error above the 50mb pressure level suggested changes to the channel set, which were incorporated in run 170.

In run 170, the 880 even numbered TIGR profiles were used for training and the 881 odd numbered TIGR profiles were used for testing the network, as before. Input to the net is brightness temperature for 728 AIRS channels, selected for surface and air temperature sensitivity, taking into account previous sensitivity analysis. Output is surface temperature and air temperature at 64 distinct pres-

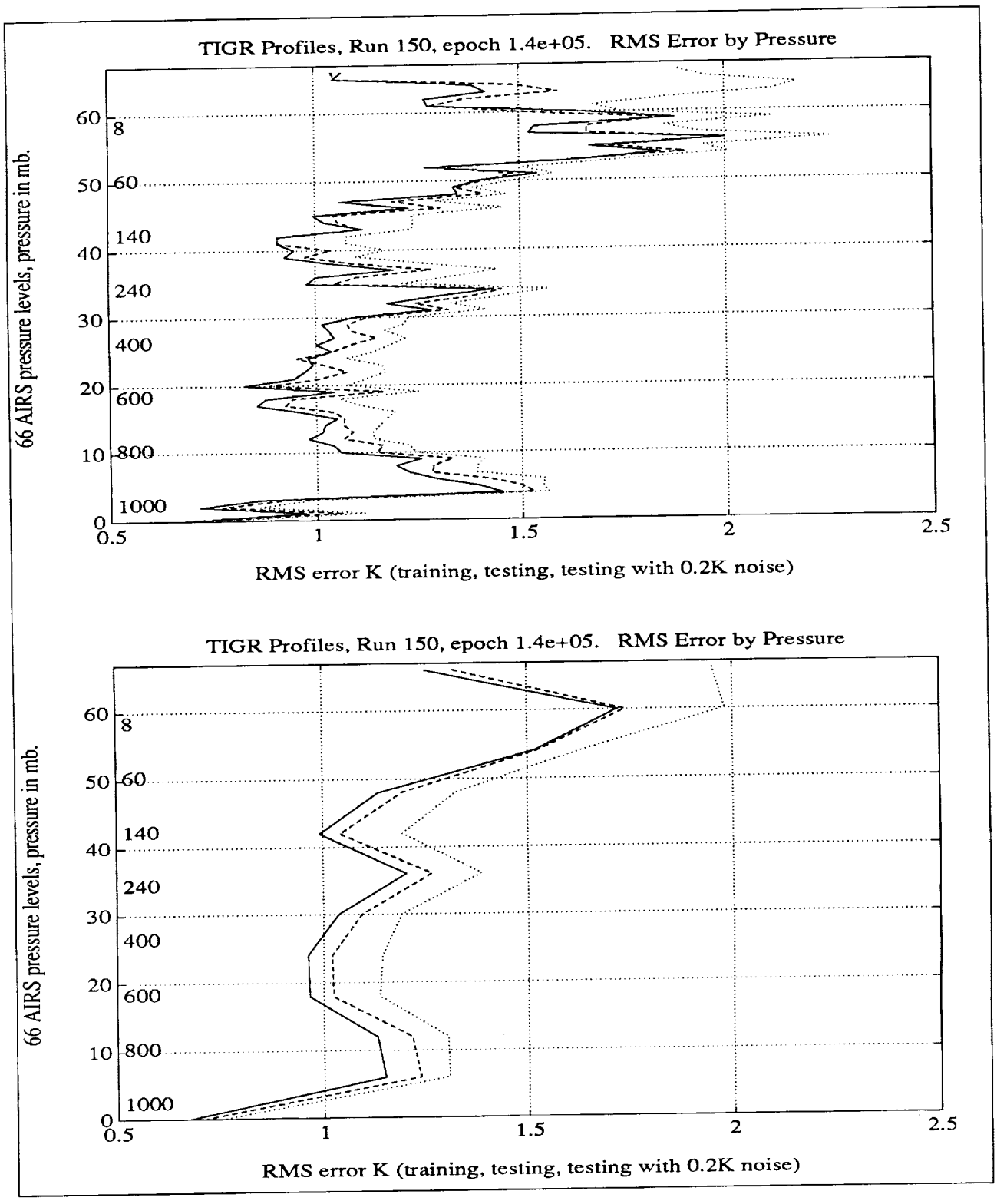

Figure 2: RMS temperature errors for run 150.

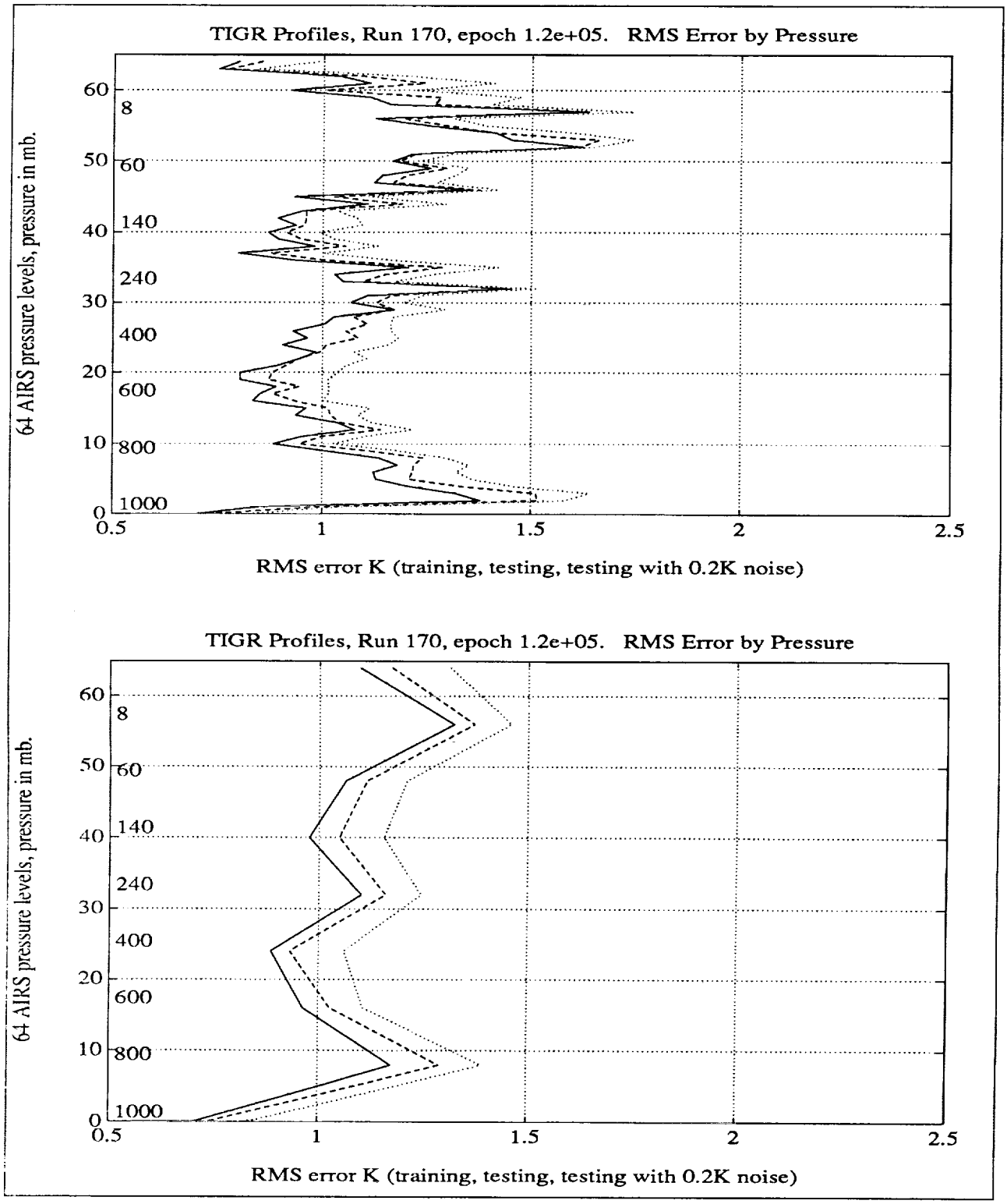

Figure 3: RMS temperature errors for run 170.

Run	Net Size	Epoch	RMS errors (a)		
			train	test	noise
150	666 × 108 × 72 × 67	240,000	1.22K	1.23K	1.37K
170	728 × 108 × 72 × 65	160,000	1.02K	1.09K	1.22K
90	586 × 90 × 60 × 50	50,000	13.2%	15.0%	15.9%

Table 1: Summary of runs discussed.

sure levels.[1] The network is the same size at the network for run 150. After 160,000 epochs, RMS training error is 1.02K, RMS extrapolation error is 1.09K, and RMS extrapolation error with 0.2K std noise is 1.22K. These results are shown in Fig. 3. A slight improvement in noise performance of this network could probably be realized by further training with noisy data.

A sensitivity analysis of run 170 is shown in Fig. 5. Note that the 'flat spot' (the large group of unused middle channels) is much reduced, but that there are still some unused channels.

Fig. 6 shows some initial results for water retrievals. Input to the net is brightness temperatures for 586 AIRS channels, selected for both water and temperature sensitivity. The same set of TIGR profiles were used as in runs 150 and 170, while the network was slightly smaller, with 90 transfer functions in the first hidden layer and 60 in the second. After 50,000 epochs, overall error for the first 50 pressure levels (expressed as percentages) is 13.2% training error, 15.0% extrapolation error, and 15.9% extrapolation error when 0.2K std noise is added.

As with more traditional methods of interpolation, neural networks can both under- and over-fit. High training error or inability to converge on the training set is a sign of underfitting, while poor performance on new data is a sign of over-fitting. The close correspondence between training and extrapolation errors on all the runs, and appropriate smoothness of retrieved profiles, suggest that the size of our hidden layers is not too large, and that we are not overfitting. It may be possible to use larger hidden layers to improve training and also (though to a lesser degree) extrapolative behavior.

6 Sensitivity Analysis

Once a network has been trained we can obtain a measure of its dependency on the input

[1] We switched from 66 to 64 pressure levels to match conventions used for the AIRS science team "write test."

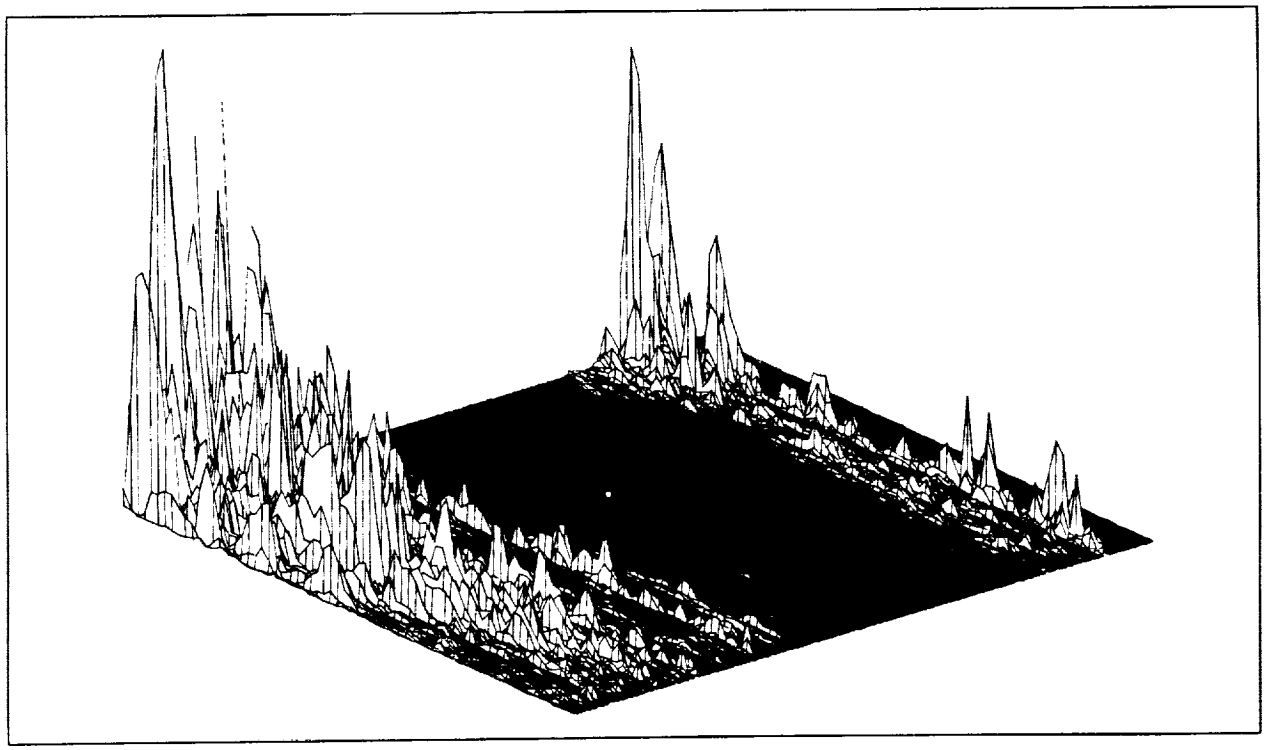

Figure 4: Sensitivity plot for run 150.

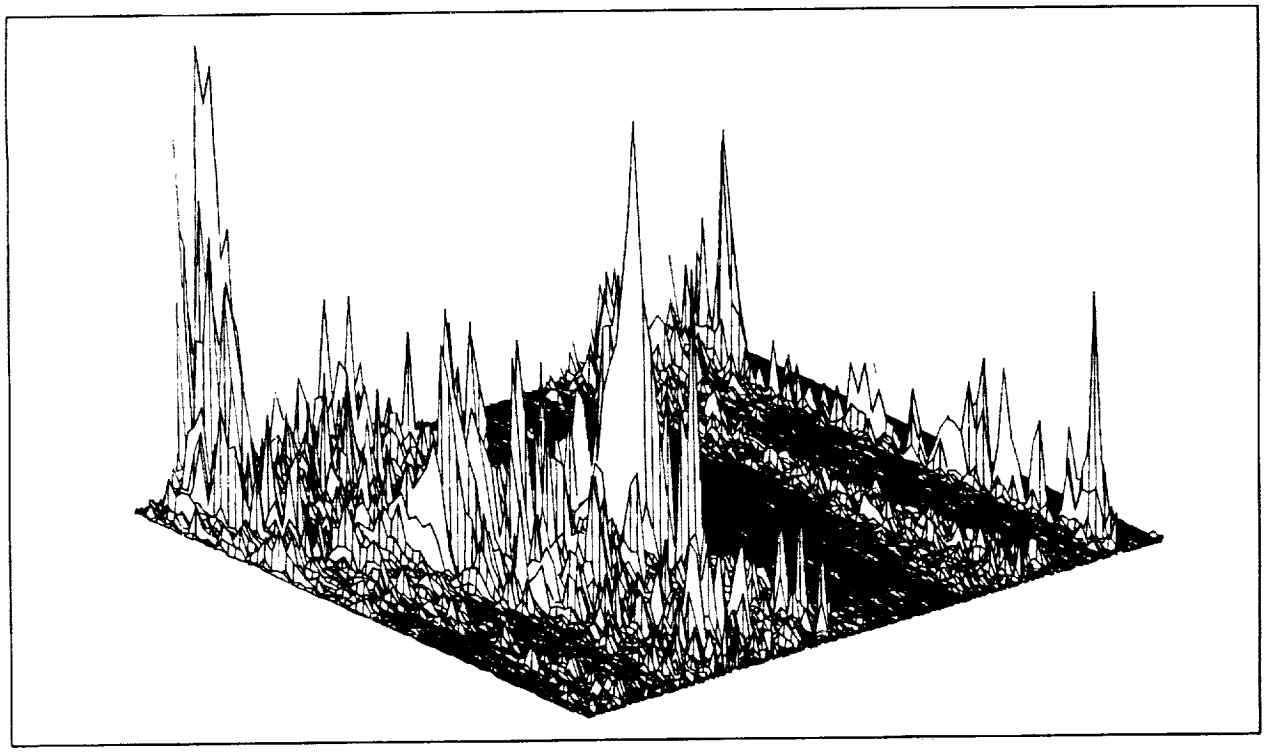

Figure 5: Sensitivity plot for run 170.

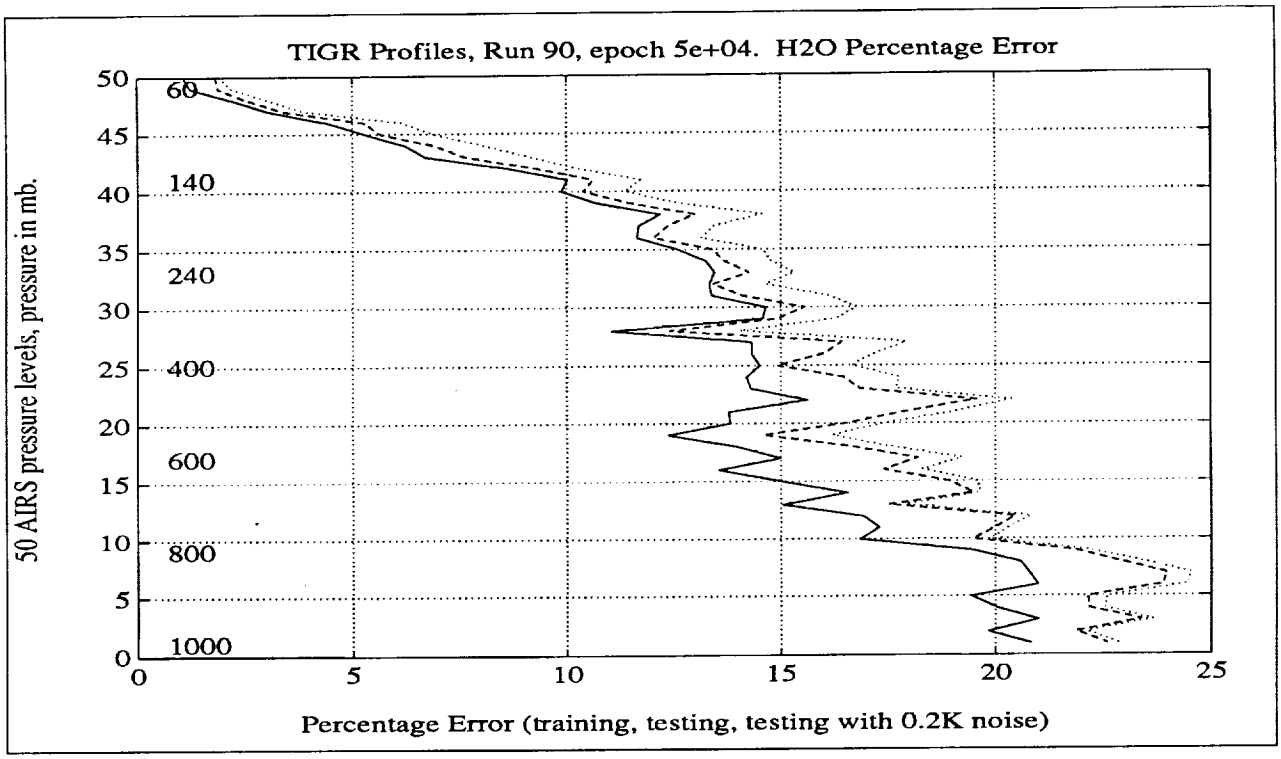

Figure 6: % errors for H_2O for run 90.

channel set by computing the Jacobian matrix of the partial derivatives of outputs with respect to inputs evaluated at a representative sample of profiles. In particular we have computed numerically by differences the quantity

$$S_{ij} = \sum_{\gamma} (\frac{\Delta T_i^{\gamma}}{\Delta \Theta_j^{\gamma}})^2 / N_{\gamma}$$

where γ indexes over the set of profiles in the dataset, N_{γ} is the number of profiles in the dataset, and Δ is the difference operator. If S_{ij} is large then on average over the set of all TIGR profiles frequency channel j has a large effect on temperature (water) in pressure layer i, while if it is small then the network has found little dependence of frequency channel j on the temperature (water) in pressure level i.

In the plots of sensitivity analysis Figs. 4 and 5, channels run from left to right, with the lower wavenumbers to the left. Pressure levels run from front to back, with the surface at the back of the plot. The z axis represents sensitivity (the sum square of partials), averaged across all the training profiles.

For many channels, sensitivity peaks correspond to weighting function peaks. The sensitivity plot looks much more 'noisy' and this is to be expected. (The sensitivity plot for an untrained net looks much like uniform noise.) In effect, the net has discovered its own representation for the weighting functions, where information from groups of channels is used to retrieve information about a particular pressure level. We conjecture that the 'noisy looking' sensitivity plot is inseparable from the network's good performance on noisy input.

7 Conclusions

We have demonstrated an application of back-propagation neural networks to the retrieval of accurate atmospheric temperature and water profiles, using the hundreds of channels of spectral information that will be available on the AIRS instrument. The prohibitive cost of training such large networks with large training sets is ameliorated by an effective mapping of the algorithm to the parallel architecture of the Maspar MP-1. The neural network allows us to make effective use of the large AIRS channel set, especially for better noise performance. Once the network is obtained it can be used to obtain very fast retrievals even with many input channels on modest computational platforms.

A sensitivity analysis of the network suggests ways we can refine the choice of channels used by the network. In principle, one could take the entire AIRS channel set, train a net for (say) temperature retrievals, perform a sensitivity analysis on the resultant net, get a smaller set of temperature sensitive channels, and use the smaller channel set to train a second net.

There are a number of directions for further work. Our present results indicate it is likely that a somewhat larger net may have errors below 1K. It may be that *simultaneously* retrieving temperature and water using a large combined channel set will give even better results than so far obtained. The retrieval of other atmospheric parameters, such as O_3, are promising areas for further investigation, as are the potential application of neural nets to cloudy atmospheres.

Acknowledgement: The authors would like to thank Alain Chedin and Milt Halem for helpful discussions of this problem.

References

[1] connections updates per second = (#weights + #biases) × #epochs × #training exemplars/ (cpu time).

[2] Water error = $\sum_i \sum_j Q_i^j / N_{levels} N_{profiles} \times 100\%$, where $Q_i^j = (1 - \frac{\text{q\_obs}_i^j}{\text{q\_calc}_i^j})$, where i is the level index and j is the profile index, and where q_obs and q_calc are observed and calculated water density in units of g/cm^2.

[3] Atmospheric Infrared Sounder: Science and measurement requirements. Technical Report D6665 Rev. 1, Jet Propulsion Laboratory, 1991.

[4] A. Chedin, N. A. Scott, C. Wahiche, and P. Moulinier. The improved initialization inversion method: A high resolution physical method for temperature retrievals from satellites of the tiros-n series. *Journal of Climate and Applied Meteorology*, 24:128–143, 1985.

[5] A. Deepak, H.E. Fleming, and J.S. Theon. *RSRM '87: Advances in Remote Sensing Retrieval Methods.* Deepak Publishing, 1988.

[6] Robert Hecht-Nielsen. *Neurocomputing.* Addison-Wesley Publishing Company, New York, NY, 1990.

[7] C. Rodgers. Retrieval of atmospheric temperature and composition from remote measurements of thermal radiation. *Rev. Geophys. Space Phys.*, 14:609–624, 1976.

[8] Patrick K. Simpson. *Artificial Neural Systems.* Pergamon Press, Inc., Elmsford, New York, 1990.

[9] J. Susskind, J. Rosenfield, and D. Reuter. An accurate radiative transfer model for use in the direct physical inversion of HIRS2. *J. Geophys. Res.*, 88:8550–8586, 1983.

[10] J. Susskind, J. Rosenfield, D. Reuter, and M. T. Chahine. Remote sensing of weather and climate parameters from HIRS2/MSU on TIROS-N. *J. Geophys. Res.*, 89:4677–4697, 1984.