Real-Time Atmospheric compensation and surface temperature estimates from Satellite using Neural Networks

Gary Bouton
CUNY City College

How does access to this work benefit you? Let us know!
Follow this and additional works at: http://academicworks.cuny.edu/cc_etds_theses
Part of the Civil Engineering Commons

Recommended Citation
Bouton, Gary, "Real-Time Atmospheric compensation and surface temperature estimates from Satellite using Neural Networks" (2013). CUNY Academic Works.
http://academicworks.cuny.edu/cc_etds_theses/639

This Thesis is brought to you for free and open access by the City College of New York at CUNY Academic Works. It has been accepted for inclusion in Master's Theses by an authorized administrator of CUNY Academic Works. For more information, please contact AcademicWorks@cuny.edu.
Real-Time Atmospheric compensation and surface temperature estimates from Satellite using Neural Networks

Thesis

Submitted in partial fulfillment of requirement of degree
Masters of Engineering (Civil)
at
The City College
of The City University of New York

By
Gary Bouton
November 2013
Approved

Professor Reza Khanbilvardi, Thesis Advisor

Prof. Julio Davalos Chairman
Department of Civil Engineering
Table of Contents

Abstract ................................................................................................................................................. 3

Introduction ........................................................................................................................................ 4

1. Procedure ......................................................................................................................................... 6

1.1. Real Time GOES and Radiosonde Retrieval ................................................................................ 8

1.2. Radiosonde Sampling Methodology ............................................................................................. 9

1.3. MODTRAN Processing and Testing ............................................................................................ 10

2. Neural Network Development ........................................................................................................ 165

2.1. Neural Network Background ....................................................................................................... 206

2.2. Neural Network Design ............................................................................................................... 209

2.3. Correction Factor ......................................................................................................................... 20

2.4. Preliminary Neural Network Tests ............................................................................................... 20

2.5. Neural Network Robustness ......................................................................................................... 254

2.6. Application to Satellite Imagery ................................................................................................... 255

2.7. Blending Satellite and Model Data ............................................................................................... 257

3. Surface Temperature Detection .................................................................................................... 33

4. Conclusion ....................................................................................................................................... 37

Code: MODTRAN TAPE 5 Reader Wrapper ......................................................................................... 38

References ............................................................................................................................................ 47
Real-Time Atmospheric compensation and surface temperature estimates from Satellite using Neural Networks

Abstract

Military satellites are being used to determine potential security threats using accurate high sensitivity measurements from defense satellites. These systems focus much more on sensitivity and Signal/Noise Ratio (SNR) than on multispectral capability. Therefore, these systems are highly vulnerable to thermal absorption and reemission that can affect the thermal signature prior to launch. In particular, there is a need to use additional assets to estimate the atmospheric temperature and water vapor profile so that an estimate of the atmospheric processes can be obtained and a correction developed to improve the thermal detection. In particular, the use of existing meteorological geostationary assets is crucial to this effort since these satellites can estimate the atmospheric temperature and water vapor profiles which in principle can be “inverted” to get the surface signature. However, this approach cannot be implemented in real time efficiently and therefore we need to develop a more empirical compensation approach which can be used in real time with minimum computer resources. In this thesis, we develop a Neural Estimator approach to take metrological inputs of water vapor and temperature over three integrated pressure levels WV1 (0.9Ps < P < Ps), WV2 (.7Ps < P < 0.9Ps), and WV3 (.3Ps < P < .7Ps) from the
existing GOES-13 sounder with additional information like zenith angle, and radiances processed from MODTRAN. The result was a robust neural network that could be applied to multiple sites and weather scenarios without much error in the ground temperature outputs. With an RMSE of .90 (K) in comparison to surface temperatures, good correlations on a near real time feed could be possible by this approach to give detection targets and provide instantaneous results to ground temperature unknowns.

Introduction

Real Time analysis on micro-scale observations in weather and climate are the latest trend in atmospheric studies as instrumentation, coverage, and modern satellite design pushes the calculable boundary of climate science. GOES-R, the newest addition to the GOES family of satellites, is set for launch in early 2015 and on board its payload boasts the latest in optical and electromagnetic instruments capable of real time retrievals at greater resolutions than on previous iterations. The technology present and in the future opens up a new field of study into micro detection of atmosphere and ground based anomalies that can lead to real time reporting and detection. The detection of ground based anomalies such as forest fire sources, ice cap calving or hot target missile launch detection requires accurate observation of the earth in the infrared. The road block in this approach is that the thermal spectral bands of interest are also obscured by water vapor contamination thus obscuring the
electromagnetic response signal of the thermal source aimed to be detected. Past approaches to correcting for water contamination in the atmosphere was to use historical models based on past climate actively which can generalize a region by its land type. In the modern environmental landscape a specific and evolving approach must be designed that grows with the system and is not reliant on an archive that is outdated to handle to modern issues we are facing. The approach of this study is to leverage real time meteorological data that is provided by the current GOES sounder to provide the needed modern data needed for water vapor correction. To achieve a learning system that can evolve and deliver in real time a neural network is to be developed that can be fed climate measurements from the GOES Satellite such as temperature and pressure profiles to the highest possible accuracy, over a very large region. Using the neural network (NN) will allow the correction to be done in near-real time.

Table of Figures

Figure 1: Python Data Retrieval Program ........................................................................................................9
Figure 2: Seasonal Temps, Water, and Radiance fluctuations...........................................................................10
Figure 3: Sensor to Target zenith angle (via Modtran manual) ...........................................................................12
Figure 4: Radiosonde measured Total Water Vapor .vs -log(tranbave) ...............................................................12
Figure 5: Radiosonde measured Total Water Vapor .vs -log(tranb2h20ave) .........................................................13
Figure 6: Radiosonde measured Total Water Vapor and Transmissions ...............................................................13
Figure 7: Radiosonde Total Water Vapor vs. Modtran redrawn H2O estimates ..................................................144
Figure 8: Seasonal variability in Inputs to NN .....................................................................................................165
Figure 9: General NN Architecture ..................................................................................................................16
Figure 10: Neural Network Design schematic showing the Temperature and Pressure components with additive MODTRAN functions ...........................................................................................................209
Figure 11: NN training (using 25% data) using all three specified temperature and water vapor levels...21
Figure 12: NN testing using only total water + 800-1000mb temp ranges.............................................................22
Figure 13: NN with added radiance factors of absorption and emission ..............................................................23
Figure 14: Surface Temp and Total WV between NYC and LA .........................................................................254
Figure 15: Neural Network run of LA using NYC local data ..............................................................................265
1. Procedure

The building and subsequent implementation of a neural network needs to ingest extensive realistic water vapor and temperature profiles at the surface pressure levels that mimic the GOES water vapor and temperature retrieval products. Training data was pulled from radiosonde launches that provide full atmospheric profile data by height, and surface pressures obtained from the archived radiosonde network archived at http://weather.uwyo.edu/upperair/sounding.html. The radiosonde data then needs to be converted to a form compatible with GOES satellite observations, therefore we need to calculate the total water columns for given pressure levels of the study. Temperature also needs to be integrated over the pressure levels recorded by height in which the radiosonde levels to fit a similar schema we wish to inject in the design of our neural network. To calculate total water vapor, first the mixing ratio of water vapor needed to be determined. The mixing ratio is defined as:

\[ MxR = \frac{P(wv)}{P(atm)} \]  

[1]
where $P(wv)$ in equation 1 is the water vapor pressure, $P(\text{atm})$ and is the atmospheric pressure. The atmospheric pressure is determined from the radiosonde and can be integrated by height, however the water vapor pressure must be further induced by calculation by:

$$P(wv) = P(\text{sat}) \times RH \quad [2]$$

Where in equation 2 RH is the Relative Humidity recorded at each sounding height and pressure level, and $P(\text{sat})$ is the saturated pressure which in turn needs to be calculated by:

$$P(\text{sat}) = 6.1708 \times e^{\frac{Tc+17.2694}{Tc+238.3}} \quad [3]$$

Where in equation 3 $Tc$ is the atmospheric temperature at the recorded height and pressure of the radiosonde reading for that day and site. With all the subsequent variables recursively solved we can then calculate the mixing ration in equation 1. With the Mixing ratio obtained for every layer of the radiosonde readings calculated the next calculation was to find the particle density of water for this has an added attenuation effect in regards to an added correction to the optical depth of the atmosphere when we are targeting a solution to an estimator to read surface temperatures. This was calculated by finding the number of air molecules in one cubic meter by equation:

$$pN(\text{air}) = \frac{N(\text{air}) \times \frac{P}{P_0}}{\frac{T}{T_0}} \quad [4]$$

Where in equation 4 $N(\text{air})$ is Avogadro's constant specific molecules of water in a layer of air in the units of particles/m$^3$ ($2.504 \times 10^{25}$ particles/m$^3$). $P$ is the atmospheric pressure and $P_0$ is the atmospheric pressure at STP, likewise $T$ is temperature and $T_0$ is temperature at STP. Having calculated the particle density of air molecules we could use the mixing ratio obtained before to calculate the number of particles/m$^3$ or water vapor density calculated as:

$$pN(wv) = pN(\text{air}) \times \text{mix} \quad [5]$$
Lastly in calculating and then converting the water vapor pressure in cm of water was to integrate the number of particles/m² calculated in the above equation, to particles/m² over the three relevant pressure bands between: Therefore, we need to train the NN based solely on partial integrated water vapor columns: WV1 (0.9Pₛ < P < Pₛ) WV2 (0.7Pₛ < P < 0.9Pₛ) and WV3 (0.3Pₛ < P < 0.7Pₛ) where Pₛ is the surface pressure which is generally estimated from the target elevation in the following equation:

\[ p \int_{z_2}^{z_1} pN(wv) \, dz \]  

[6]

The calculated integrated layers in the calculated data is in particles/m² and was then converted to units in cm of water by virtue of the fact that 1 cm column of water = 3.36×10²⁶ particles.

1.1. Real Time GOES and Radiosonde Retrieval

Python code was developed to retrieve and store the most recent 30 minute GOES 13 and 15 data from the University of Wisconsin’s (CIMSS) satellite database that correlated with in situ metrological radiosonde launch data we were collecting from [http://weather.uwyo.edu/upperair/sounding.html](http://weather.uwyo.edu/upperair/sounding.html). This process cut the human error and effort of manually retrieving and searching through archives to find time matchups with the radiosonde launches themselves. When the call is made in the Python code to search the GOES archive it checks the local directory to see if the latest file already exists to avoid duplication. Then the retrieval Python code downloads updated files that the directory is missing matching it with the dates and hour of the latest meteorological data itself which makes processing faster and more accurate.
in reducing redundancy. A snapshot of the program itself is shown below.

Figure 1: Python Data Retrieval Program

1.2. Radiosonde Sampling Methodology

It is of great importance that to train a robust neural network the sampling input used to train the estimator itself must be as varied and representative of the system it must predict. The issue with current ways to estimate surface anomalies is that the archive system is not robust only providing feedback for
large general systems than the granular specific feedback system a more accurate estimator needs. To provide the neural network a wide array of meteorological inputs seasonality that mimics the variety and variation that can naturally occur in climate systems an estimator would encounter. Figure 2 illustrates the importance of seasonal variation in the training data because the temperature and water vapor profiles are component over the atmosphere (TOA) monochromatic radiances for different seasons. The side bar of radiances derived from radiosonde temperature/water profile (blue spectrum) versus the summer US standard climatological models (at 40N Lat) (green spectrum) highlights this meteorological seasonal deviations does have an impact on the end point radiometric properties that would be used in forecasting and estimating the ground target in our modeling system. Large errors in estimated outputs can be corrected by accounting for this noise in the training data as seen here.

1.3. MODTRAN Processing and Testing

To calculate the large number of profiles pulled for training we employed MODTRAN 5 to process the high resolution spectrally resolved transmission factor. MODTRAN itself had to be rewritten to be able
to process the large number of atmospheric layers that the radiosonde themselves measure. This involved the writing of a Python script that changed the MODTRAN .ini file to increase its input caps on user provided atmospheric profiles and to provide a folder read/write data logging script to change the default startup of MODTRAN to load the user made CARD 5. There was no MODTRAN wrapper for running multiple user provided atmospheric profiles at the time so a series of scripts were need to then handle this process from a manual one to a scheduled automatic job. A CARD 5 development tool was built in Matlab that turned the radiosonde files from the Python managed local directory from their text format into a MODTRAN 5 readable format writing the atmospheric vertical layers to be processed and the attributes at which MODTRAN would run the spectral processing at. From here, the band averaged transmissions can be obtained from the spectral transmission $T(v)$ as;

$$T_j = \frac{\int_{-\infty}^{\infty} [F(v)I(v)P_{surf}(v,T_z)] dv}{\int_{-\infty}^{\infty} [F(v)P_{surf}(v,T_z)] dv}$$

[7]

Where the filter function for the band is estimated as a simple rectangular response and the surface emission is estimated from the black body spectrum at the radiosonde surface temperature. The radiances were output for 2 water absorbing channels at wavenumbers between the STG band – the water band (i.e. 2.7-3.0 microns) and 3.0 – 3.2 microns. The zenith angle of the instrument seeing the ground target was also modeled and varied in the data at angles of 0, 30, and 45 degree angles.
Zenith angle importance in the model is variable because the line of sight from instrument to ground target could pass through more or less vertical columns of water vapor just based on the spread area of the sight angle at which the focal point is at.

Initial testing of the MODTRAN results was necessary because of the customization made to the root MODTRAN processing .ini files and to the TAPE 5 wrapper which processed the radiosonde data into readable formatting. The testing was done on water vapor behaviors as a function of the radiances that MODTRAN calculated from the input.
Figures 5, 6 and 7 exhibit the log relationship you would expect between increasing water vapor and transmission in the thermal channel being tested. Figure 7 displays the MODTRAN re-estimate of water
vapor being fed into its algorithm. The MODTRAN program creates layer estimates of the total water that may differ than input because it interpolates missing atmospheric layers from the data. As Figure 7 illustrates there is a very good relationship with estimate and actual measured total water.

Figure 7: Radiosonde Total Water Vapor vs. Modtran redrawn H2O estimates
2. Neural Network Development

Neural Network development is part science, part statistics, and part creativity in that the inputs, and the variability of those inputs can greatly affect how good an estimator the NN is for a given output. Some input variables may greatly enhance an NN product and reduce estimation error, while some variables may act as noise and actually create larger divergence from the output than expected. It is important to understand the variability in your input and how they may change as a function of time over a number of inputs on a time scale. Therefore if your training variables have a greater variance over time then your training data must also represent this change as you train the neural network. For instance in this project using variables of just air temperature at pressure levels and water vapor density at pressure levels for just the New York City area alone we find great variability through the calendar year for these measurements.

![Figure 8: Seasonal variability in Inputs to NN](image)

As can be seen from figure 8, there is great seasonal variability and the neural network must be trained with this in mind or else we could expect larger errors when developing our program to give us expected outputs for thermal anomalies if the neural network is given optical depth and the known water vapor
density in a column of air. Therefore when creating our training data we randomly sort the data to provide a realistic representation of the variability of known data points in to which the neural network can create a proper error regression from the distance of points of the data to each other as a function of the time of the year and the level at which the measurement is taken at in the vertical air column.

2.1. Neural Network Background

NN models attempt to reconstruct the inverse relationship between the spectral measurements of the satellite and the unknown surface temperature measurement. The architecture of the one-hidden layer NN used is shown in Figure 9.

![General NN Architecture](image)

We can build a NN with multiple hidden layers, but we choose not to, since more layers make the examination of its internal structure more difficult. Furthermore, it is proven that the one-hidden-layer NN is able to model any continuous function which is suitable for our case.

The first layer $S_0$ is the input layer, which contains the observations $Y$. The last layer $S_2$ is the output layer that generates the aerosol properties $X$. The hidden layer $S_1$ contains “nodes” that function as...
building blocks of the associations between the inputs and the outputs. What comes into every node \( k \) is the sum of the weighted inputs plus a bias term \( (\sum_i w_{ik} Y_i + b_k) \) and what comes out of it is the transfer function of this sum. In our NN this transfer function is the hyperbolic tangent sigmoid function \((\text{tanh})\). The NN output \( X \) is the weighted sum of the hidden layer node output augmented by a bias term. More specifically, the output \( X_j \), where \( j \) denotes one of the retrieved aerosol properties, is calculated from the equation:

\[
X_j = \sum_{k \in S} w_{kj} \sigma(\sum_{i \in S} w_{ik} Y_i + b_k) + b_j
\]  

[8]

where \( \sigma \) is the transfer function, \( w \) denotes the NN weights and \( b \) the NN biases. The weights and biases express the associations between the input and the output. Thus, after the training of the NN, we can retrieve the aerosol parameters by simply applying the above equation on the observations of the total intensity and linearly polarized intensity of light. Therefore, although the time required to build the NN is usually long, after the training is done, the retrieval of surface temperature from a combination of satellite band inputs and parameterized atmospheric properties is instantaneous.

2.1.1. Neural network training

We train the NN with the “resilient backpropagation algorithm” which is available from the MATLAB neural networks toolbox\(^1\). The algorithm calculates numerically the weights and biases of the NN that minimize the total root mean square error (RMSE) \( E(w, b) \) between the NN estimated values and the true values of the aerosol properties in the training dataset:

\[
E(w, b) = \frac{1}{2N} \sum_{s=1}^{N} \sum_{j \in S} D(\hat{X}^s_j, X^s_j)^2
\]  

[9]

\(^1\) We selected this particular training algorithm among all the available ones in the MATLAB neural networks toolbox, after testing them all and evaluating their speed and the performance of the NNs they build.
where $s$ denotes one of the $N$ samples of the training dataset, and $D$ is the Euclidean distance. During the training stage the resilient backpropagation algorithm optimizes the weights and biases accordingly to the sign of the gradient $\frac{\partial E}{\partial w_{ik}}$. For each iteration $w_{ik}$ is increased or decreased by an adaptive step $\Delta w_{ik}$ as follows:

$$w_{ik}(t + 1) = w_{ik}(t) + \Delta w_{ik}(t)$$  \hfill [10]

with

$$\Delta w_{ik}(t) = \eta_{ik}(t) \cdot \Delta_{ik}(t - 1)$$  \hfill [11]

where $t$ is the number of the iteration and the product $\eta_{ik} \cdot \Delta_{ik}$ defines the adaptive step of the particular weight. The step takes an initial value of $\Delta_{ik}^0$, which is increased or decreased in every iteration by $\eta_{ik}$, depending on the change of the gradient $\frac{\partial E}{\partial w_{ik}}$ sign. If the sign remains the same, the convergence is in a shallow region and for speed-up purposes the step is increased by $\eta_{ik} = 1.2$. If the sign changes, there has been a jump of a minimum to the previous step, thus the step is decreased by $\eta_{ik} = -0.5$. The training stops, when $E(w,b)$ reaches a predetermined value, or when there is no more improvement on it. At that point the inverse function, defined by the calculated NN weights and biases $(w,b)$, estimates the aerosol properties of the training dataset with the least possible total RMSE for the particular NN configuration.

2.1.2. Neural network testing

---

2 The particular weight $w_{ik}$ is used here to represent any of the weights and biases of the NN, since similar formulas apply to all of them.
The performance of the constructed NN is evaluated with a testing dataset. This can be comprised of simulated or real data. In our research, our NN was comprehensively tested with simulated measurements (chapter 4), as well as available real measurements from the RSP (chapter 5). The obvious need for testing with simulated cases, is that it can provide an estimation of the output uncertainties of the NN, since the true values of the aerosol parameters are known. A more thorough discussion about this approach follows in the next section.

2.2. Neural Network Design

We need to ingest realistic temperature and pressure profiles obtained from measurements and models, i.e. water vapor and temperature profiles from radiosondes and GOES 13 soundings. And then to perform full forward radiative transfer modeling using GENSPECT Code. We then train the neural network against the computed GENSPECT for effective optical depth at our 3 temperature, and water vapor density levels which are functions of their respective pressure levels.

Figure 10: Neural Network Design schematic showing the Temperature and Pressure components with additive MODTRAN functions
In figure 10, the architecture for training the attenuation factor is illustrated. Included in the NN training are the three water vapor and temperature layers discussed above as well as the surface pressure $P_0$ and the target pressure $P_T$.

2.3. Correction Factor

The correction factor we need is the effective optical depth of the atmosphere which we define as the band averaged power at satellite divided by band averaged power at the ground due to the source. Clearly, this definition includes the effects of atmosphere absorption and re-emission. This is calculated as:

$$\tau_{eff} = \frac{\int P_{TOT}(\lambda) F^i_{sen}(\lambda) d\lambda}{\int P_g(\lambda) F^i_{sen}(\lambda) d\lambda}$$

[8]

2.4. Preliminary Neural Network Tests

When training a neural network it is common practice to use about 25% of your data to train a proper model assuming the points picked are randomly selected and representative of your total population. The other 75% of the data is for model validation to test for errors and to improve the NN weighting schema if there are large variation on the model edges. The first set of tests were set up to test the neural network variables to obtain a known output by using a variety of inputs.
The above figure 11 highlights what you would expect of a neural network responding in a proper linear relationship with the training set data in which we calculated and what the output is predicted as from the neural network training. We used the traditional 25% of forward processed radiative data calculated by GENSPECT for two band ranges in the thermal range (2.7 – 3.0 microns, and 3.0 – 3.2 microns) and also using our technique of using 3 levels of data from temperature and water vapor that we discussed.
earlier.

Figure 12: NN testing using only total water + 800-1000mb temp ranges

Figure 12 highlights an important use case of how using select data not representative of the physical make of the environment you are trying to represent with a neural network can lead to very erroneous distance issues. In regards to getting near exactness to expected outcomes in terms of aerosol optical depth as we are using it here for we need to be stringent on the data going in as input and testing different multiples from our data gives a richer understanding on how the NN actually responds. We disregard all pressure ranges except the near surface readings with regards to using only the total column of water vapor in air which produces greater error bars.
Figure 13: NN with added radiance factors of absorption and emission

Figure 13 is an illustration how small performance and improvements can be made to an NN model by adding new features to the calculated results from MODTRAN. Here we run radiances in both absorption and absorption and re-emission modes with the latter giving a much tighter reference and boundary to the longer thermal band being used in testing. In some cases in using an NN as predictor a tighter spread is easily more desirable then one in which the tails of the model fall further from the means of the clusters themselves.
2.5. Neural Network Robustness

To test the robustness of our neural network we used similar radiosonde data of collected pressure information levels as we did before collecting air temperature and calculating for water vapor density and then seeing if the neural network could respond correctly in an environmental setting that is climatologically different than our own.

![Figure 14: Surface Temp and Total WV between NYC and LA](image)

We can see from figure 15 that variables that are important to our study such as near surface air temperature and total water vapor is different in NYC than in California. While California is considerably warmer year round it has overly seasonally much less total water vapor in the air, being a drier locale.
We note that even when the neural network was used that was trained over New York and is applied to data in California; good correlations are still obtained proving the robustness of the neural network approach.

### 2.6 Application to satellite imagery.

Once the NN approach has been tested on individual locations, we can explore the use of this approach using the satellite imagery from GOES-15. In figure 16, we plot the images for the temperature and water vapor profiles as obtained from the GOES-Sounder retrieval algorithm from CIMSS. This data is available every 30 minutes allowing a quick real time estimation of the correction factor.
Figure 16: GOES-15 (West) observations of the three water vapor and temperature layer products. Left column is WV in mm of water and the right column are the three temperatures.
2.7 Blending Satellite and Model Data

Unfortunately, satellites have cloud issues that can effect the result so we also blend the satellite data with meteorological profiles from the NOAA’s Rapid Refresh meteorological model. Once the NN is trained, real time access to the atmospheric data is needed. The strategy is to use the GOES sounder retrievals of $T_j$, $W_j$ (j=1:3) to ingest directly into the NN input stream. The other orbital and scenes are obtained directly from the satellite and surface characterizations based on USGS data. It must be emphasized that the cloud contamination is a significant problem in these retrievals. In our case, since our application to assess thermal retrieval of ground sources only clear sky observations are suitable, using only the satellite cloud clear data is most likely sufficient. However, even in clear sky or in the case of thin cirrus clouds, the GOES retrievals may not be made in cases where the HEO sensor can pick up suitable signals. In this case, the satellite data must be supplemented with additional data that we take from suitable near real time 1 hr. weather forecasts. In particular, NOAA’s Rapid Update approach which has come online May 1, 2012 allows us to fill in the cloud obscured data with forecast data. Therefore, combining these 2 data sources will provide the atmospheric state variables needed in the NN processing. Of course, in developing this blended database, a hierarchy must be established regarding the best date to be used. In figure 17, we plot an example Rapid Refresh model output over the entire CONUS region.

Figure 17: Rapid Refresh 800-1000mb temperature
To blend these 2 data sources, we follow the following hierarchy.

1) GOES data takes precedence

2) The Rapid Refresh data fills the holes

3) To make the data consistent, we plan to use a multivariate regression between the GOES and Rapid Refresh retrievals for the points where both data are available to remove biases between the model and the satellite retrievals.

a) GOES Data Ingestion: In figure 15, a GOES retrieval of the three layer averaged water vapor and temperature sounding is made for a winter case. Note for example, the reasonable results for the mountain and coastal areas as well as ocean warming.

b) Rapid Refresh (RR). The model is the state of the art providing high-frequency updated (every 1h) short-range weather model forecasts (out to 18h)

To assess Rapid Refresh data was retrieved at different sites across the continental United States. Their values were aggregated, interpolated and compared with Radiosonde data. The following sections describe the results retrieved from the analysis. The following maps present the retrieved Radiosonde values on top of the available Rapid Refresh data. The Radiosonde data was retrieved at 1200 UTC. The Rapid Refresh data presented is the assimilated data and the forecasted data generated on 04/18/2013 at 1800 UTC forecasting forward 18 hours to 1200 UTC. The entire data set represents forecasts that point to 04/19/2013 1200 UTC. In figure 18 and 19, we overlay the radiosondes for both surface and PBL averaged temperature while figure 20 and 21 plots the regression performance. The results show that even for an 18 hr forecast, performance of the model is quite impressive and is therefore a useful approach to providing estimates needed for atmospheric compensation.
Figure 18 - Forecasted Surface Values (Forecast Hour = 18)

Figure 19 - Forecasted PBL Values (Forecast Hour = 18)
Figure 20 – Correlation of Forecasted Surface Values (Forecast Hour = 18)

Figure 21 – Correlation of Forecasted PBL Values (Forecast Hour = 18)
The ingestion of the RUC analysis and surface pressure into the NN leads to the following near real time map for the transmission factors for the 2 thermal bands as seen from the DoD defense satellite as plotted in figure 22. We note that the 2 transmission bands are generally consistent with higher transmissions over the mountain regions where less water vapor absorption is expected.
3. Surface temperature detection

In the previous section, we use the NN architecture to look primarily at the atmospheric transmission factor. This approach ignores for the atmosphere reemission and also does not efficiently use the concept of the NN as an architecture to efficiently reproduce the inverse function. As discussed earlier, the above estimator for band averaged extinction is most useful when the source radiance is much larger than the background. However, any validation efforts must be done in cases which are more terrestrial. In particular, targets such as water bodies or homogenous surfaces offer the best approach to calibrate thermal sensors. Therefore, a full Radiative Transfer approach must be used taking into account not only the atmospheric profile data but the land surface data. The general structure of the NN approach is given in the flow chart of figure 23.
Figure 23: Neural Network model used for validation of Radiative transfer models and NN reduction.

The forward modeling now comprises the following steps.

1) Use the MODTRAN5 Radiative transfer code to calculate band averaged spectral Radiance

\[ R_j = \frac{\int \int [F(\nu)] R(\nu) d\nu}{\int [F(\nu)] d\nu} \]

over the appropriate response functions. To calculate this quantity, the following atmospheric and land surface parameters are in general needed.

a. A continuous water vapor profile \( w(p) \) as function of pressure

b. A continuous temperature profile \( T(p) \) as function of pressure

c. The surface emissivity \( \varepsilon_j \) (assumed constant over the band).

d. The surface pressure (p0)

e. The geometric view angle. \( \theta_v \)

f. Possible Aerosol Contamination which is quantified using an aerosol class and a quantitative measure of opacity (i.e. Aerosol Optical Depth at 500nm) \( \tau_{500} \)
2. Again, to ensure real time processing, the NN is developed based solely on partial integrated water vapor columns and layer temperatures but now the emissivity as well as the observation angles as well as an aerosol visibility are all allowed to vary statistically over a realistic distribution of uncorrelated values.

Applying the Neural Network modeled to a near real time surface temperature estimator involved the building of a delivery package that tested and shuffled input data to remove bias from seasonality or time order. MODIS data was used as the test satellite for atmospheric climate data that would correlate well.

*Figure 24: Test and Sample results for Surface Temp NN*

if GOES but at a much broader spectrum. 25% of the data was sampled for training with recorded ground temperatures as the target output and the inputs being the zenith angle, the three water vapor integrated pressure levels, with the same pressure levels for temperature, the optical depth, and the absorption and emission radiances from MODATRAN calculations. Early results were good for the sampled MODIS data test for a surface temperature predictor with an RSME of .86 with only small errors being attributed to a large range in emissions from 0.9 to 1.0 due to some of the larger differences.

The next test was to now use GOES 13 retrieved profiles to test and run the NN for accuracy against the preliminary results when MODIS instruments were used.
The final test of GOES 13 retrieved atmosphere retrieved water vapor, press, and temperature levels produced very solid results at a RMSE of .90. This result showed not many large errors in the way of random spikes or noise but of note was the larger errors were seen near freezing temperatures.
4. Conclusion

The neural network estimator was developed to handle 2 different cases. The first case was the development of an effective attenuation parameter directly from meteorological observations. The results were quite promising for both channels. In addition, we saw very clearly that the total water vapor is not a good estimator for atmospheric attenuation and the addition of vertical resolution for both temperature and water vapor was crucial for strong correlation performance. In the second application, the NN was trained not only with atmospheric data but the satellite band outputs as well as solar geometry and surface emissivity to get an estimate of the surface temperature. The approach for surface temperature proved to be not only accurate but robust in handling data points that were not originally part of the training set locations. The complexity of the neural network was reduced to only 2 hidden layers and it still performed quite admirably even when pushed with data that can be seen as seasonably and locally variable in terms of atmospheric metrological components which may be only specific to one region. The wide reach of the NN itself allows this technique to be applied in real-time across many different lat/lon and time scenarios without the need to generalize a region, or landscape that can enable a very bad estimation of the surface temperature parameters.
Code: MODTRAN TAPE 5 Reader Wrapper

[datsonde,txt]=xlsread('inputsonde.xlsx');

%Card 1 Development (REQUIRED)  Main Radiation Transport driver

MODTRAN = 'M'; %set to T or M band model

SPEED = 'S'; %set for speed of k - correlations S for slow upper atmosphere, M for medium, 17 k values

BINARY = 'F'; %all outputs in ASCII format

LYMOLIC = ''; % do not include auxiliary species

MODEL = 7; % 7 Sets atmospheric model to user supplied data eg. radiosonde
ITYPE = 2; % sets geometric path for ;line of sight, 1 sets horizontal path 3 sets vertical to ground

IEMSCT = 2; % 0 Program executes in spectral transmittance only mode
% 1 Program executes in spectral thermal radiance (no sun / moon) mode
% 2 Program executes in spectral thermal plus solar / lunar radiance mode

IMULTI = 0; %sets multiple scattering, 0 executes program without multiple scattering

M1 = 0; M2 = 0; M3 = 0; M4 = 0; M5 = 0; M6 = 0; MDEF = 0; %for user supplied profiles Card 2C1 (radiosonde)

IRD = 1; % set to 1 when user data is to be supplied

NOPRNT = 0; %set to 0 for normal writing to tape 6 & 7

TPTEMP = .000; %sets boundary temps

%SURREF = ' '; %albedo of the earth set to 1

SURREF = 'LAMBER'; %Spectral Lambertian surfaces are specified by CARD 4A, % 4L1 and 4L2 inputs.

Card1 = {MODTRAN, SPEED, BINARY, LYMOLIC, MODEL, ITYPE, IEMSCT, IMULTI, M1, M2, M3, M4, M5, M6, MDEF, IRD, NOPRNT, TPTEMP, SURREF};

% Card 1A Development (REQUIRED) Radiative transport

DIS = 'f'; % set to f because IMULTI is 0, no multi scattering
DISAZM = ' '; % set for false to use only visible fluxes

DISALB = ' ';

NSTR = 8; % number if strings 8 is default

SFWHM = 0; % dont know wavenumbers

CO2MX = (380.000); % CO2 mixing ratio ppmv

H2OSTR = '1.00000'; % default water column

O3STR = '1.00000'; % default ozone column

CPROF = '0';

LSUNFL = 'f'; % use spectral resolution of modtran band model

LBMNAM = 'f';

LFLTNM = 'f';

H2OAER = 'f';

CDTDIR = 'f';

SOLCON = 0; % zero as default

CDASTM = ' ';
ASTMC = 0;

ASTMX = 0;

ASTMO = 0;

AERRH = 0;

NSSALB = 0;

Card1A = {DIS, DISAZM, DISALB, NSTR, SFWHM, CO2MX, H2OSTR, O3STR, CPROF, LSUNFL, LBMNAM, LFLTNM}; %H2OAER, CDTDIR}; %, SOLCON, CDASTM, ASTMC, ASTMX, ASTMO, AERRH, NSSALB};

% Card 2 (REQUIRED)  CLOUD OPTIONS and AEROSOLS

APLUS = ' '; % is blank for default

IHAZE = 0; % No cloud or aerosol interaction

CNOVAM = ' '; % default

ISEASN = 0; %season determined by input

ARUSS = ' '; % blank is default

IVULCN = 0; % volcanic is default

ICSTL = 3; %default is 3
ICLD = 0; % no clouds or rain

IVSA = 0; % set to 0 = not used

VIS = 0; % default set by IHAZE

WSS = 0; % wind speed default settings

WHH = 0; % wind speed default settings

RAINRT = 0;

ML = length(datsonde(:,1)); % find # of atmospheric levels

IRD1 = 0; % no read
IRD2 = 0; % no read

Te = 'NY GOES-sonde_30 deg';

l=1;
for w=1:2:(ML-1)
    l(w)=++1;
    g=sum(l);
end
MLr=round(ML/2);
Card2C= {MLr, IRD1, IRD2, Te}; % Not clear why gary is counting every other layer???????
Card2C = \{ML, IRD1, IRD2, Te\}; % reading every layer

% CARD2c1 user supplied radiosonde

\[ ZM = \text{datsonde}(;1); \text{\%Height in km} \]
\[ P = \text{datsonde}(;2); \text{\%Pressure} \]
\[ T = \text{datsonde}(;3); \text{\%Goes Temp} \]
\[ WMOL1 = \text{datsonde}(;4); \text{\%Water vapor (Expressed as Dew Point)} \]
for \( t=1:(ML) \):
    \[ WMOL2 = .0; \]
    \[ WMOL3 = .0; \]
    \[ Pt = 'A'; \text{\% pressure in mbar} \]
    \[ Tt = 'A'; \text{\% temperature in K} \]
    \[ Wvt = 'F'; \text{\%Water vapor (Expressed as Dew Point)} \]
    \[ \text{unit2} = '666666666666'; \]
    \[ \text{Card2C1}(t,:) = \{ZM(t), P(t), T(t), WMOL1(t), WMOL2, WMOL3, Pt, Tt, Wvt, unit2\}; \]
end

GNDALT = ZM(1);

Card2 = \{APLUS, IHAZE, CNOVAM, ISEASN, ARUSS, IVULCN, ICSTL, ICLD, IVSA, VIS, WSS, WHH, RAINRT, GNDALT\};

%%% CARD 3 (REQUIRED) LINE OF SIGHT
H1 = ZM(ML); % Initial altitude (km)
H2 = GNDALT; % Final altitude (km)
Zenith = 180.000; % (35deg) Initial Zenith angle as measured from H1
Card3 = {H1, H2, Zenith, .000, .000, .000, 0, 0.0000};

%% CARD 3A1
IPARM = 12; %Controls the method of specifying the solar/lunar geometry
IPH = 2; %Selects Mie-generated internal database of aerosol phase
 %functions for the MODTRAN® models
IDAY = 32; %Day of the year from 1 to 365 used to specify the earth to sun
 %distance and (if IPARM = 1) to specify the sun's location in the sky
ISOURC = 0; %Extraterrestrial source is the sun
Card3A1 = {IPARM, IPH, IDAY, ISOURC};

%% CARD 3A2
PARM1 = -174.219; % relative solar azimuth (degrees east of north)
PARM2 = 53.8; % solar zenith (degrees)
Card3A2 = {PARM1, PARM2};

wavemin=1000;
wavemax=5000;

Card4 = {wavemin, wavemax, 1, 2,'r','m', 'w2aa'};
Card5 = {0};

% end

%% CARD 4A
NSURF = 1; %Determined from the atmospheric temperature profile
AATEMP = 0; %Determine it from the atmospheric temperature profile
DH20 = 0; %Liquid water option, water layer thickness input [mm]
MLTRFL = 'F'; %Embedded surface moisture attenuation model
Card4A = {NSURF, AATEMP,DH20,MLTRFL};

%% CARD 4L1: defines the name of the input data file being used to
% define the spectral albedo.
SALBFL = 'DATA/spec_albSGP.dat';
Card4L1 = {SALBFL};
% SALBFL: Name of the spectral albedo data file. The default
% spectral albedo file, 'DATA/spec_alb.dat' may be used or a
% user-supplied file. If a user-supplied file is specified, it
% must conform to the format described in the default file.

%% CARD 4L2: defines the number or name associated with a spectral
% albedo curve from the SALBFL file.
CSALB = 'SGP';
Card4L2 = {CSALB};

file2 = 'Radiosondetest_surface.tp5';
fid1 = fopen(file2,'wt');

fprintf(fid1,'%s%s%s%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%i%
fprintf(fid1,'%2s%3d%4d%3s%2d%5d%5d%5d\%10.5f%10.5f%10.5f\n',Card2{:});
fprintf(fid1,'%5d%5d%5d\n',Card2C{:});
for j = 1:(ML)
    fprintf(fid1,' %\%10.3E %\%10.3E %\%10.3E %\%10.3E %\%10.3E %\%10.3E%s%s%s%s\n',Card2C1{j,:});
end;
fprintf(fid1,'%10.3f%10.3f%10.3f%10.3f%10.3f%10.3f%5i %10.5f\n',Card3{:});
fprintf(fid1, '%5d%5d%5d%5d\n', Card3A1{:});
fprintf(fid1, '%10.3f%10.3f\n',Card3A2{:});
fprintf(fid1,'%10.0f%10.0f%10.0f%10.0f%s%s %s\n',Card4{:});
    fprintf(fid1, '%1d%9.2f%9.3f%c\n', Card4A{:});
    fprintf(fid1,'%s\n', Card4L1{:});
    fprintf(fid1,'%s\n', Card4L2{:});
fprintf(fid1,'%d\n',Card5{:});

close(fid1);
% file3 = sprintf( 'Radio%d.tp6', n );
% fid2 = fopen(file3,'wt');
% file4 = sprintf( 'Radio%d.tp7', n);
% fid3 = fopen(file4, 'wt');
% fclose(fid2);
% fclose(fid3);
strf = fileread(file2);       %# read contents of file into string
strf = strrep(strf, 'E+00', 'E+0');        %# Replace wordA with wordB
strf = strrep(strf, 'E-00', 'E-0');
fidf = fopen(file2, 'w');
fwrite(fidf, strf, '*char');       %# write characters (bytes)
fclose(fidf);

References


