CSDL Information Technical Note No. 4

GFS VS. ETA12 FORECAST MODEL WIND COMPARISONS: MONTHLY ANALYSIS AND PROGRAM DOCUMENTATION

Silver Spring, Maryland
September 2005

U.S. DEPARTMENT OF COMMERCE
National Oceanic and Atmospheric Administration
National Ocean Service
Coast Survey Development Laboratory

CSDL Informal Technical Note No. 4

GFS VS. ETA12 FORECAST MODEL WIND COMPARISONS: MONTHLY ANALYSIS AND PROGRAM DOCUMENTATION

Philip H. Richardson
Richard A. Schmalz, Jr.

September 2005

U.S. DEPARTMENT OF COMMERCE

National Oceanic and Atmospheric Administration

National Ocean Service
Coast Survey Development Laboratory

NOTICE

CSDL Informal Technical Notes present work in progress or summaries of results that are not appropriate to be published as either formal NOAA Office of Coast Survey Technical Reports or the less formal Technical Memoranda. Results are intended primarily for use within CSDL. Scientific review of the material is minimal, and CSDL makes no warranty as to its validity or completeness.

TABLE OF CONTENTS

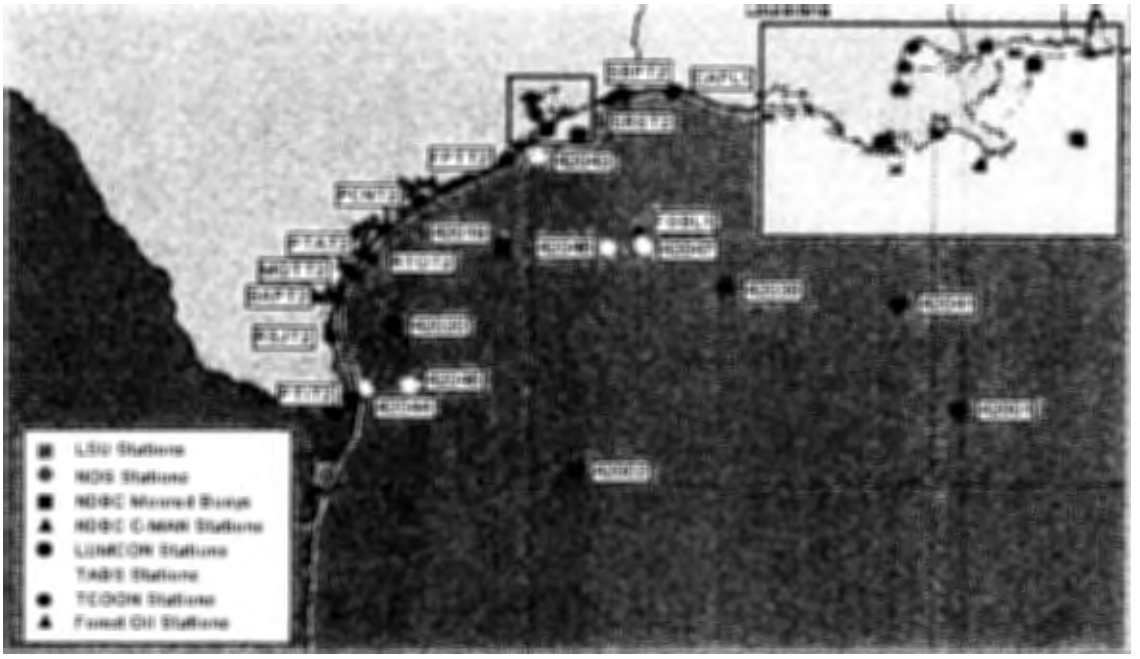
LIST OF FIGURES

LIST OF TABLES

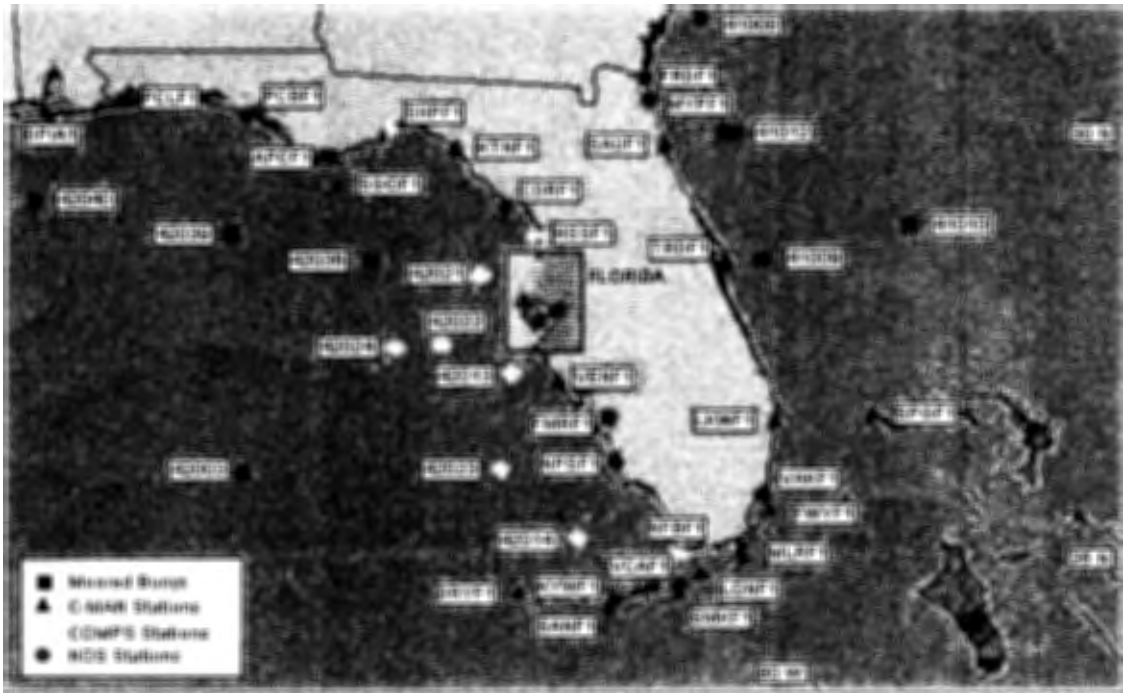
LIST OF COMPUTER PROGRAM LISTINGS

ABSTRACT

Comparative analyses of NWS/Eta12 and NWS/GFS forecast model forecasts versus observations at 12 locations around the Gulf of Mexico were performed. Monthly comparisons are presented for November 2002, January 2003, May 2003, and July 2003. Individual forecast comparisons for events are also demonstated. The Eta12 and GFS model winds compared favorably to the observations, with the Eta12 winds of superior quality for all months and especially during Hurricane Claudette in July 2003. Sea level atmospheric pressure forecasts were of near equal quality. Comparison software is documented via program descriptions with listings in Appendix A and a description of the analysis procedure in Appendix B. Script and program input files are given in Appendix C. This report is a companion to the subtidal water level comparison report by Richardson and Schmalz (2004) and allows an independent analysis of the wind and pressure forcing prior to the subtidal water level analysis. A brief outline of future work is presented to conclude the report.

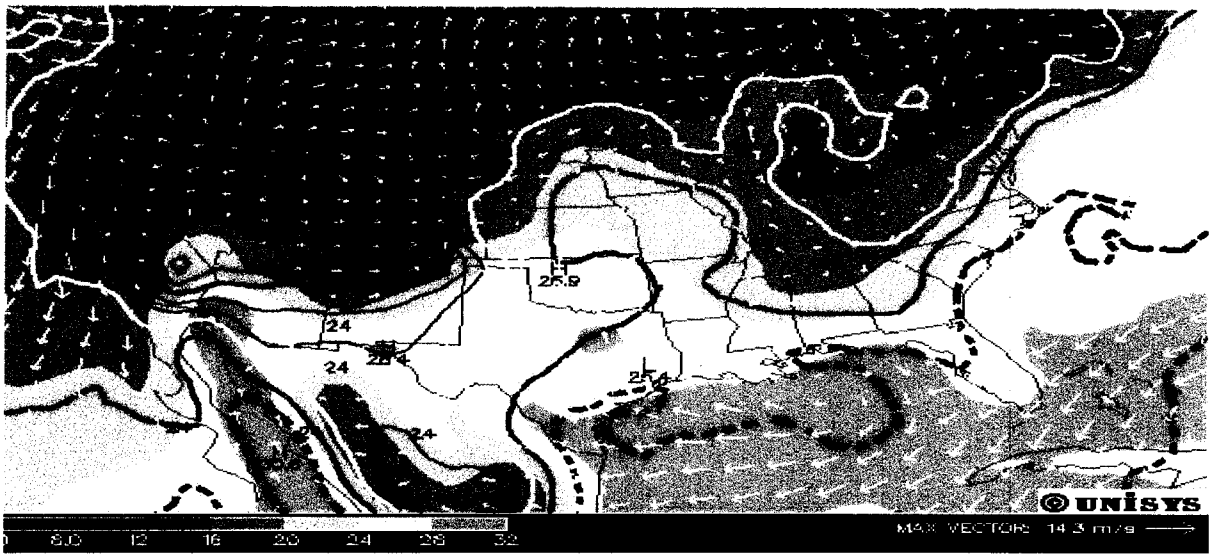


Western
Gulf (Note 42035 is located at the blue square above 42043)



Eastern Gulf
Base Map, Station Locations

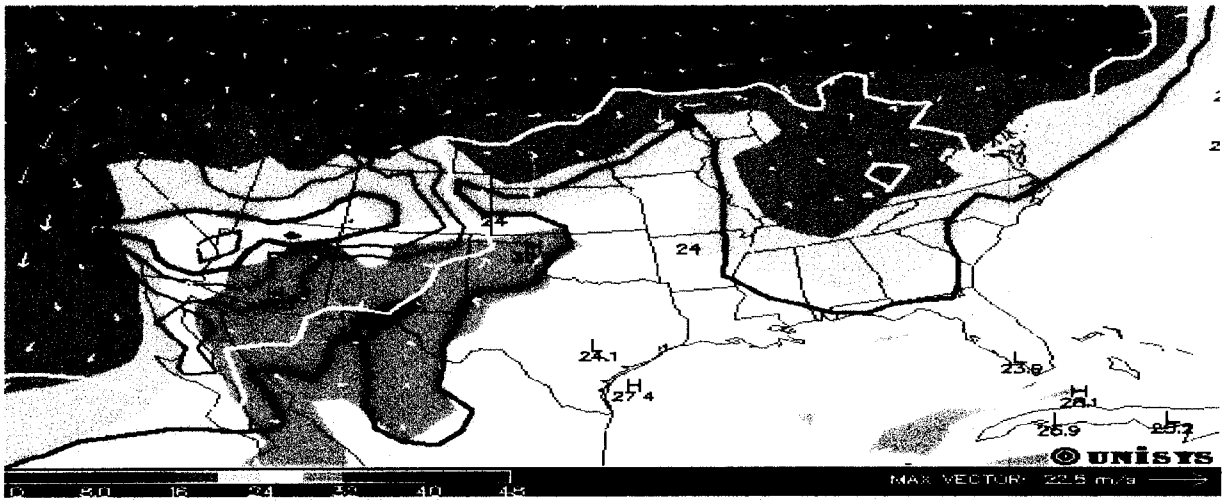
Eta 6Z 19 September 2005



Surface parameters plotted are 2 meter above ground temperature in Celsius (in color contours, see color bar), convergence (black lines, interval=4, shaded > 0), 2 meter above ground dewpoints in Celsius (colored lines, interval=5, bold orange=20, bold white=15, bold red=0, bold magenta=-15, bold gray=-30) and winds plotted as white vectors.

GFS 12Z 19 September 2005

The 1000 mb chart also shows near surface weather conditions. The parameters plotted are 1000 mb temperature in



Celsius (in color contours), convergence (black lines, interval=2, shaded > 0), 1000 mb dewpoints in Celsius (colored lines, interval=5, bold orange=20, bold white=15, bold red=0, bold magenta=-15, bold gray=-30) and winds plotted as vectors.

1. INTRODUCTION

This report serves as a companion to the 2004 report, by Richardson and Schmalz, entitled "ETSS vs. DGOM Model Water Level Comparisons: Program Documentation and Monthly Analysis ". The Dynalysis Gulf of Mexico Model (DGOM) uses US Navy COAMPS wind forcing (Patchen and Blaha, 2002), while the NWS Extratropical Storm Surge Model (ETSS) uses NWS Global Forecast System (GFS) wind forcing (Chen et al., 1993). The present report was developed in an effort to evaluate the wind and atmospheric pressure forcing prior to the analysis of the basin scale hydrodynamic model water levels. The objective of this effort was to determine the best meteorological forcings which all hydrodynamic models might subsequently use, rather than selecting meteorological forcings based on organizational convenience. With this approach, the basin scale hydrodynamic models would use a uniform surface wind stress formulation, rather than adjusting several stress relations to observe the best water level fit.

The Eta12 atmospheric forecast model is run 4 times per day out to 84 hour with a horizontal spatial resolution of 12km on a semi-staggered Arakawa E grid and uses a silhouette-step topography or Eta vertical coordinate. Further details may be found in Black (1994). More recent results are given in Pielke (2001) and in Mesinger (2000). It is considered to be the core North American Mesoscale or NAM model.

GFS, formerly referred to as the Aviation Model, uses a spectral triangular 254 and Gaussian grid of 768x384, which is roughly equivalent to 0.5 x 0.5 degree latitude/longitude. The vertical domain is represented by 64 nonuniform sigma levels, with 15 levels below 800 hPa and 24 levels above 100 hPa. For more details see the GFS Atmospheric Model (2003) website. It is run four times per day out to 84 hours.

Two new programs, forc.aviEta.f and plot_wndanal.pro, have been developed to compare the performance of forecast model windfields throughout the Gulf of Mexico. The programs were used to compare daily forecast winds, hours 6-36, from both the GFS model and the Eta12 model to observed winds at a number of stations throughout the Gulf. The statistical analysis is performed by forc.aviEta.f, which calculates both daily and cumulative (monthly) statistics. The statistical values calculated include the rms error, relative error, the bias, gain, the correlation coefficient, and the standard error. Plot_wndanal.pro is written in the IDL programming language. The program will plot the observed windspeed along with points representing the high, low, start and end points for each daily forecast. Symbols used to represent these points are plus, square, triangle, and asterisk. Plot.wndanal.pro generates one plot per page. Forc.aviEta.f is written in FORTRAN 77, while plot_wndanal.pro is written in IDL. Both programs are run on CBBAY in Linux and are documented in Appendix A with a description of both programs provided as well as program listings. In Appendix B instruction is provided to perform the analysis. Complete script and control file listings are given in Appendix C.

Since it is intended that this windfield analysis be performed first prior to analysis of the water levels

(see Richardson and Schmalz, 2004), which are generated by basin scale forecast models relying on these forecast windfields for meteorological forcing, we present the analysis results for the same months used in the water level comparison report mentioned above. Analysis results for November 2002, January 2003, May 2003, and July 2003 are presented separately in Chapters 2-5 In Chapter 6, some conclusions are drawn from the work already completed, as well as recommendations for future subjects of study.

2. NOVEMBER 2002 ANALYSIS

The observed wind plot in Figure 2.1 indicates a range of windspeed from 0 to 15 m/s. In general, there seems to be a duration of about 2 to 3 days for high windspeed events. The interval between windspeed events is about 4 to 7 days.

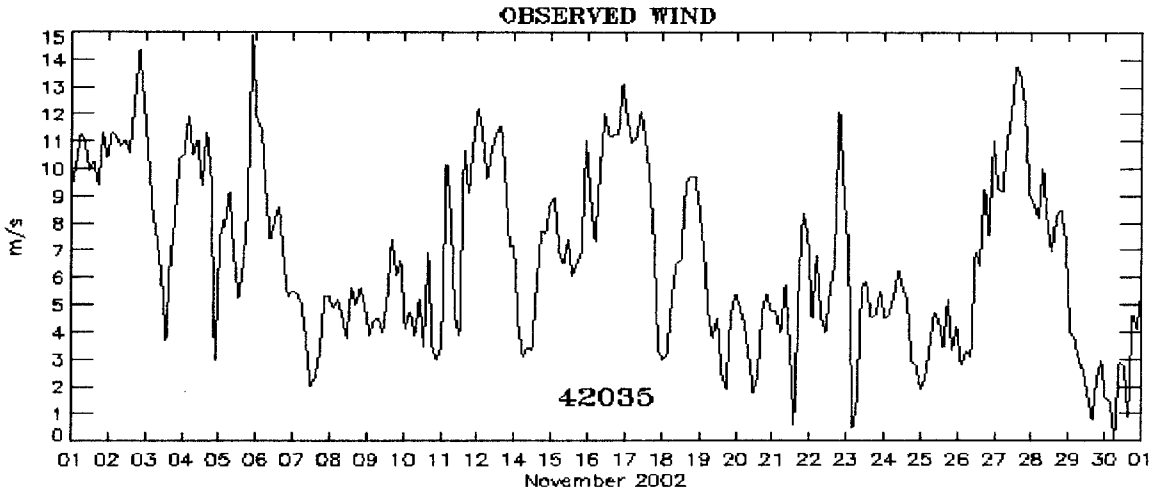


Figure 2.1 Observed Wind at Station 42035, November 2002

The statistical values from Table 2.1 indicate that Eta12 performs substantially better, in the 00z forecasts, than GFS at all of the four key stations. The four key stations are 42035, 42020, 42001, and 42036. Buoy station 42035 is located at the entrance to Galveston Bay, 42020 is located in the western gulf, 42001 is located in the central gulf, and 42036 is located in the eastern gulf. The rms error of the model windspeed to the observed windspeed is rmse1. The rms error of the model windspeed in the direction of the observed wind is rmse2. The comparison of model windspeed in the direction of the observed wind, the dot product, is the more meaningful test. Rmse3 is the rms error of the atmospheric pressure.

The statistical values from Table 2.2 indicate that the results are mixed for the 12z forecasts. One observes that the atmospheric pressure errors for Eta12 and GFS are of comparable size on the 12z forecast, where as the Eta12 is truer to observations than the GFS on the 00z cycle. Since atmospheric pressure errors tend to translate into windspeed errors, windspeed errors are comparable for both models for the 12z forecast. Eta12 does better at 42020 and at 42001, while GFS does better, for both the windspeed and the dot product, at 42035 and at 42036.

Table 2.1 Wind Statistical Analysis : November 2002, 00z forecast

Rmse1 is the rms error of the model windspeed to the observed windspeed. Rmse2 is the rms error of the model windspeed in the direction of the observed wind. Rmse3 is the rms error of the atmospheric pressure.

	GFS			Eta12		
Stations	rmse1(m/s)	rmse2(m/s)	rmse3(mb)	rmse1(m/s)	rmse2(m/s)	rmse3(mb)
42035	3.592	5.404	2.468	2.326	2.725	1.424
42019	2.665	3.480	1.262	2.113	3.042	1.748
42020	3.262	4.142	2.334	2.213	3.194	2.197
42002	4.929	6.758	3.205	2.282	2.953	1.248
42001	2.201	3.886	1.596	1.984	2.489	1.185
42041	2.354	2.722	0.978	2.676	3.073	1.596
42040	2.963	4.441	2.005	2.336	2.846	1.675
42039	2.864	3.473	1.895	1.929	2.526	1.592
42036	2.610	3.202	1.734	2.009	2.729	1.398
42003	2.193	3.139	1.698	2.077	2.473	1.324
SRST2	5.791	5.212	3.262	3.020	2.754	1.344
PTAT2	3.613	3.398	1.316	3.556	3.369	1.401

Table 2.2 Wind Statistical Analysis : November 2002, 12z forecast

	GFS			Eta12		
Stations	rmse1(m/s)	rmse2(m/s)	rmse3(mb)	rmse1(m/s)	rmse2(m/s)	rmse3(mb)
42035	3.041	4.730	3.187	3.152	4.981	3.609
42019	3.146	4.571	3.023	3.212	4.900	3.643
42020	4.197	6.029	4.068	3.535	4.684	3.892
42002	5.267	8.355	4.838	3.583	5.624	3.117
42001	3.260	6.290	3.234	2.872	4.644	2.622
42041	2.791	5.061	2.775	3.251	5.425	3.077
42040	2.867	4.289	2.986	3.223	5.295	3.191
42039	3.089	4.690	3.088	3.029	5.556	3.040
42036	2.888	4.925	3.047	3.062	5.443	2.880
42003	3.116	6.089	2.908	2.715	4.482	2.310
SRST2	5.643	4.953	3.611	3.082	3.263	3.576
PTAT2	3.768	3.610	3.181	3.547	3.406	3.653

In Figures 2.2 and 2.3, the forecast windspeed is in the direction of the observed wind. It can be seen that the GFS forecast has the greater number of outliers.

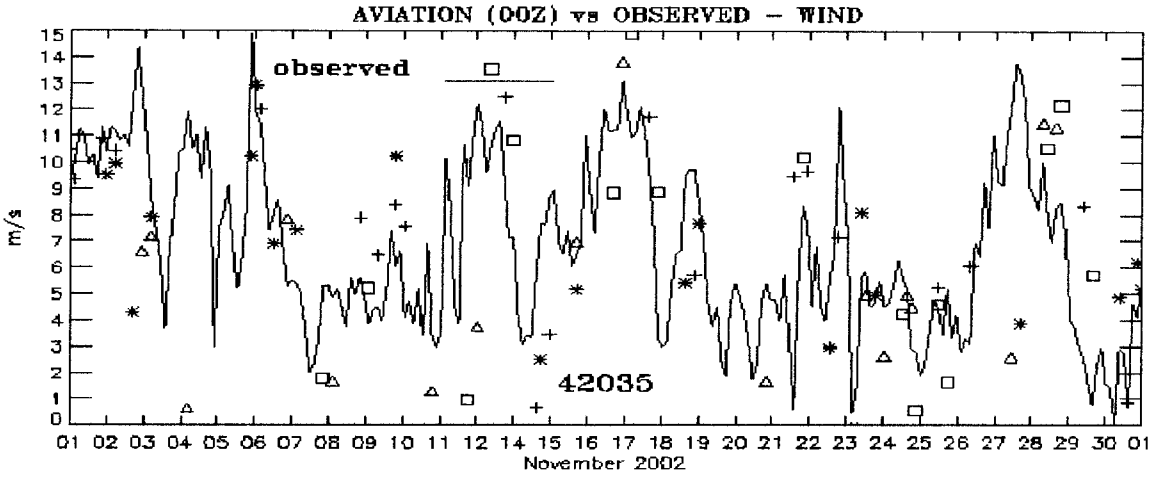


Figure 2.2 GFS (00z) vs Observed Wind at Station 42035, November 2002

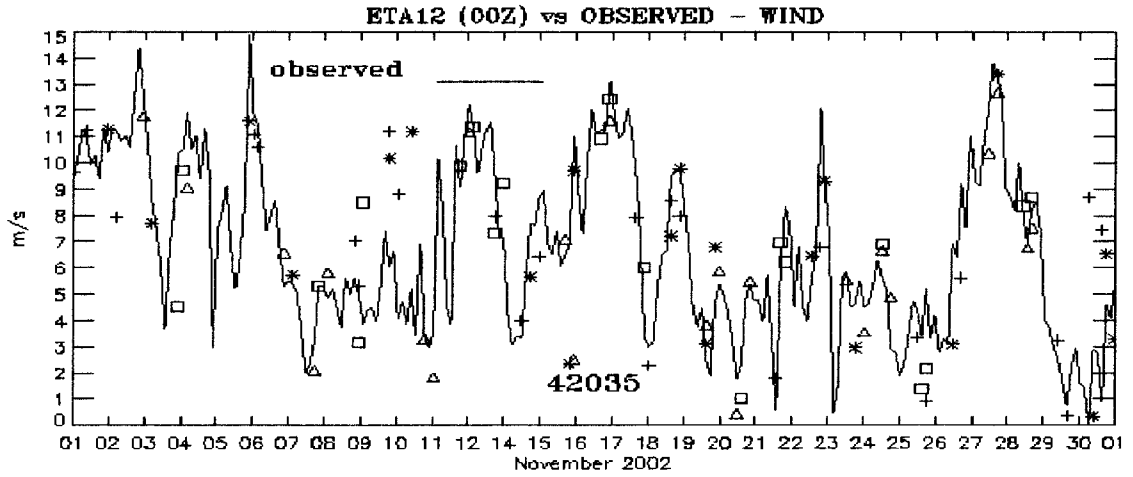


Figure 2.3 Eta12(00z) vs. Observed Wind at Station 42035, November 2002

Note : Four points are plotted using a common symbol for each daily forecast. The four plotted points include the high, low, start, and end point. Symbols used to represent forecast points include the plus, square, triangle, and asterisk.

Table 2.3 Forecast Wind Evaluation, 00z forecast, November 2002

Station	GFS npf	Eta12 npf
42035	6	24
42020	4	26
42001	6	24
42036	9	21

Note : npf is the number of preferred forecasts based on rmse.

Table 2.3 indicates that Eta12 performs much better than GFS in terms of npf. Eta12 is the better forecast, overall, about three quarters of the time.

3. JANUARY 2003 ANALYSIS

The observed wind depicted in Figure 3.1, for January 2003, is similar to the observed wind of November 2002. The range of windspeed is from 0 to about 15 m/s, and the duration of high windspeed events appears to be about two days.

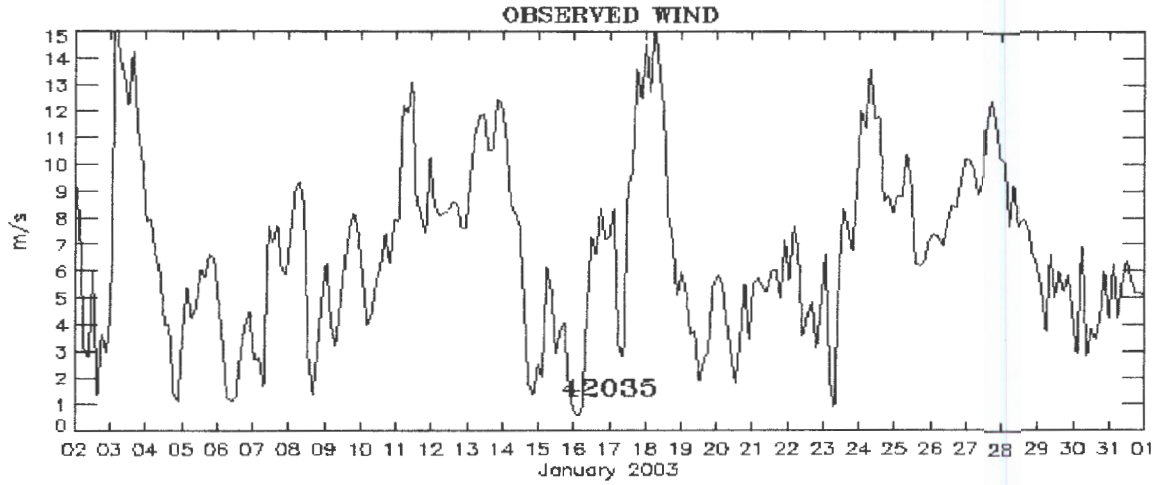


Figure 3.1 Observed Wind at Station 42035, January 2003

The analysis for January 2003 includes 00z forecasts only, since only the 00z ETTS forecasts were available for the water level analysis. Table 3.1 indicates that Eta12 performs better at all four key locations. The difference at station 42035 is particularly significant. Looking at the plots in Figures 3.2 and 3.3, which compare forecast windspeed in the direction of the observed wind with the observed windspeed, would seem to confirm this. The GFS forecast would appear to have the greater number of outliers. Table 3.2 indicates, once again, that Eta12 provides the better daily forecast most often. At station 42035, Eta12 provides the better forecast 26 times to only four for GFS.

Table 3.1 Wind Statistical Analysis : January 2003, 00z forecast

Rmse1 is the rms error of the model windspeed to the observed windspeed. Rmse2 is the rms error of the model windspeed in the direction of the observed wind. Rmse3 is the rms error of the atmospheric pressure.

	GFS			Eta12		
Stations	rmse1(m/s)	rmse2(m/s)	rmse3(mb)	rmse1(m/s)	rmse2(m/s)	rmse3(mb)
42035	3.200	4.360	2.632	1.851	2.061	1.358
42019	2.799	3.136	1.139	2.208	2.653	1.569
42020	3.262	3.767	2.151	2.620	3.138	1.952
42002	3.842	5.138	3.157	1.854	2.193	1.069
42001	2.394	2.916	2.215	2.038	2.392	1.154
42039	2.659	3.712	1.697	1.648	1.785	1.415
42036	2.185	2.900	1.443	1.746	1.821	1.319
SRST2	5.090	4.431	3.492	2.688	2.451	1.273
PTAT2	3.647	3.118	1.175	3.181	2.770	1.344

Table 3.2 Forecast Wind Evaluation, 00z forecast, January 2003

	GFS	Eta12
Station	npf	npf
42035	4	26
42020	8	22
42001	6	24
42036	8	22

Note : npf is the number of preferred forecasts based on rmse. Also, Eta12 missed one forecast for January, 2003.

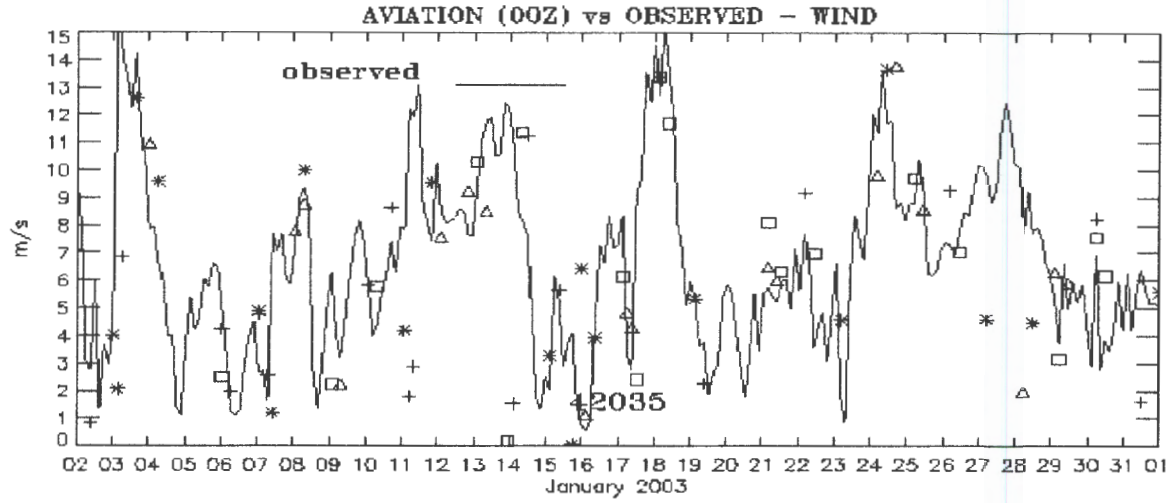


Figure 3.2 GFS (00z) vs. Observed Wind at Station 42035, January 2003

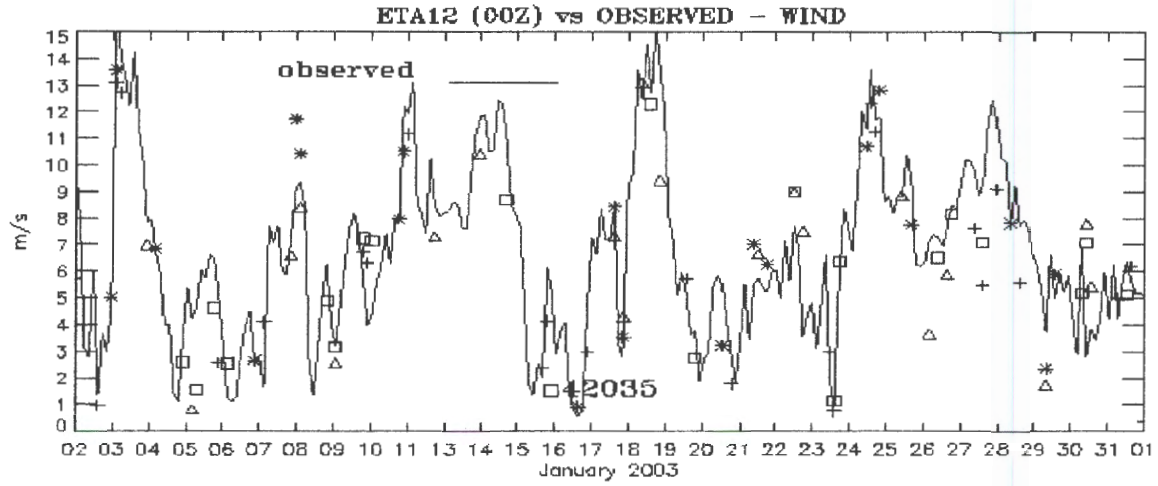


Figure 3.3 Eta12 (00z) vs. Observed Wind at Station 42035, January 2003

Note : Four points are plotted using a common symbol for each daily forecast. The four plotted points include the high, low, start, and end point. Symbols used to represent these forecast points include the plus, square, triangle, and asterisk, respectively.

4. MAY 2003 ANALYSIS

The observed wind plot in Figure 4.1 depicts a range of windspeed from 0 to about 12 m/s. May is a bit more quiescent than November or January, hence, lower windspeeds. This plot also indicates the diurnal variability of the windspeed.

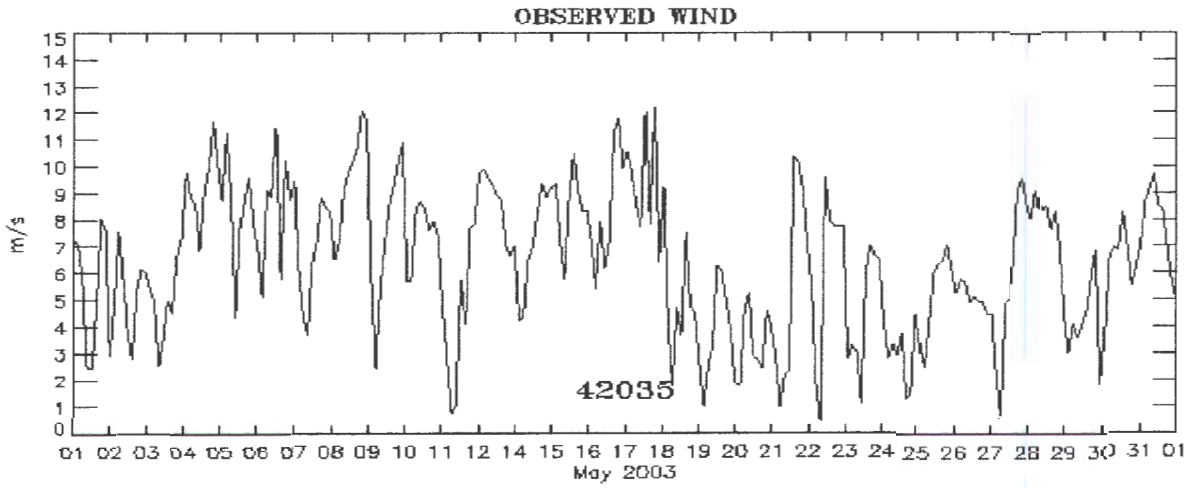


Figure 4.1 Observed Wind at Station 42035, May 2003

The analysis for May 2003 focused on the 00z forecasts only, since only these forecasts were available for the ETSS water level analysis. Table 4.1 indicates, once again, that Eta12 performs better at all four station locations. The difference at station 42001, located in the central gulf, is especially noteworthy. Looking at the plots in Figures 4.2 and 4.3, both the GFS model and the Eta12 model appear to have outliers. During the last two forecast cycles, GFS has some extreme outliers, while Eta12 holds to the observed signal quite well.

Table 4.1 Wind Statistical Analysis : May 2003, 00z forecast

Rmse1 is the rms error of the model windspeed to the observed windspeed. Rmse2 is the rms error of the model windspeed in the direction of the observed wind. Rmse3 is the rms error of the atmospheric pressure.

	GFS			Eta12		
Stations	rmse1(m/s)	rmse2(m/s)	rmse3(mb)	rmse1(m/s)	rmse2(m/s)	rmse3(mb)
42035	2.622	3.695	1.343	2.260	2.705	1.225
42019	2.006	2.126	1.213	2.277	2.420	1.502
42020	2.444	2.695	1.826	2.048	2.088	2.282
42002	4.106	5.660	1.707	1.687	1.967	1.134
42001	2.263	3.440	0.927	1.751	1.943	1.113
42041	1.727	1.914	0.842	1.898	2.229	1.335
42039	2.616	3.762	0.962	1.899	2.482	2.482
42036	2.363	3.148	1.067	1.759	2.204	2.204
SRST2	3.522	3.252	1.506	2.434	2.429	2.429
PTAT2	2.618	2.153	1.058	3.438	2.805	2.805

Table 4.2 Forecast Wind Evaluation, 00z forecast, May 2003

	GFS	Eta12
Stations	npf	npf
42035	6	25
42020	7	24
42001	1	30
42036	9	22

Note : npf is the number of preferred forecasts based on rmse.

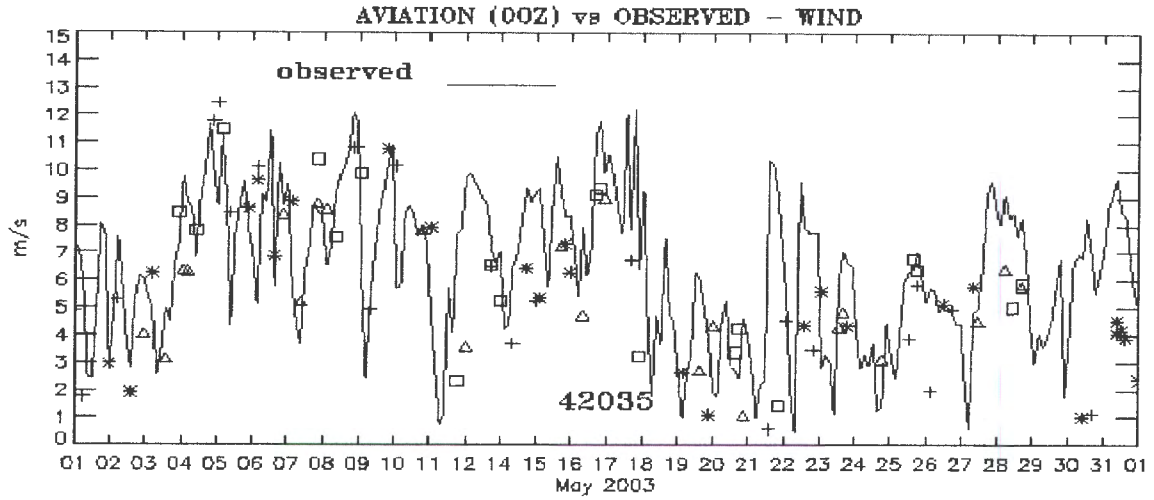


Figure 4.2 GFS (00z) vs. Observed Wind at Station 42035, May 2003

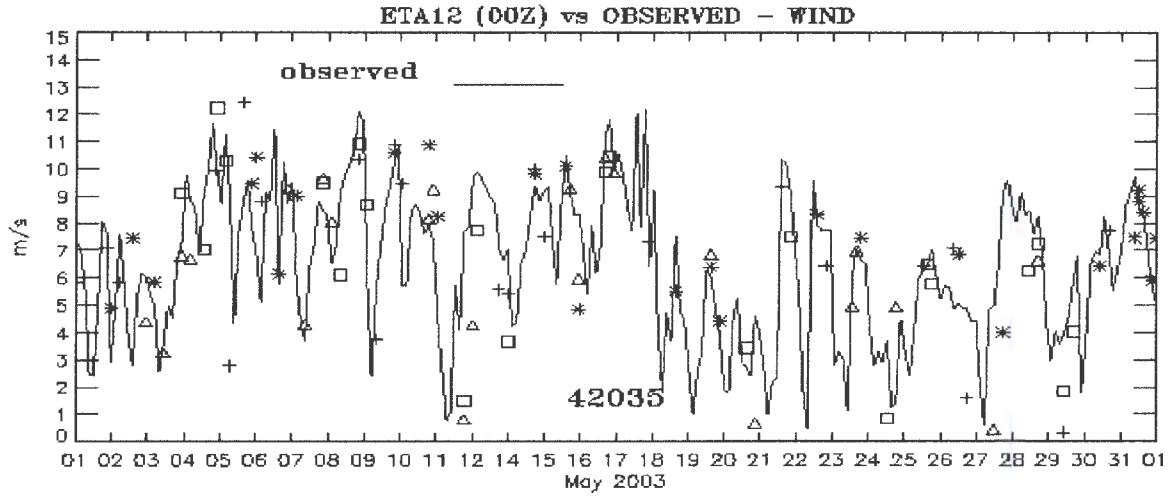


Figure 4.3 Eta12 (00z) vs. Observed Wind at Station 42035, May 2003

Note : Four points are plotted using a common symbol for each daily forecast. The four plotted points include the high, low, start, and end point. Symbols used to represent these forecast points include the plus, square, triangle, and asterisk, respectively.

5. JULY 2003 ANALYSIS

July 2003 has a greater range of windspeed than previous months. This is due to hurricane Claudette, with a maximum windspeed of close to 20 m/s. The remainder of the month is largely quiescent.

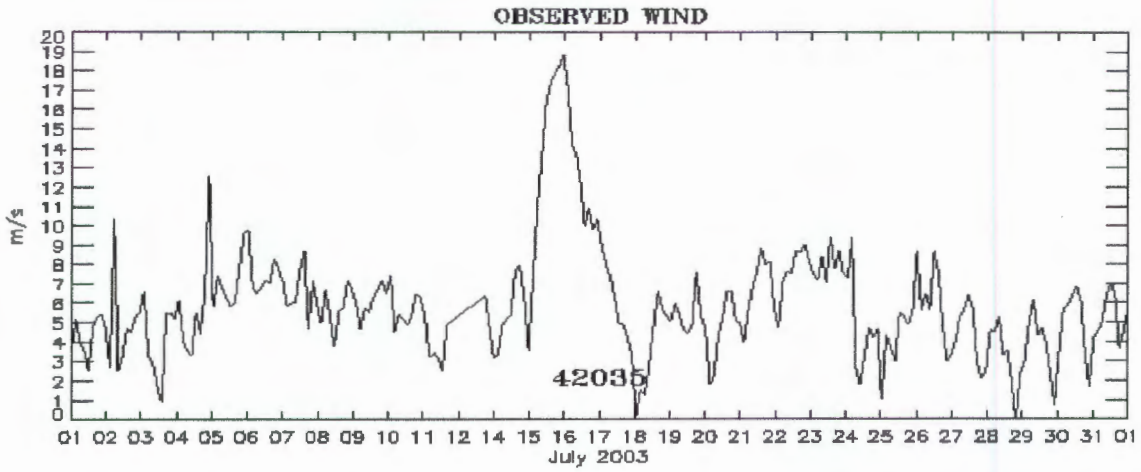


Figure 5.1 Observed Wind at Station 42035, July 2003

The analysis for July 2003 again includes 00z forecasts only, since the ETSS water levels were analyzed only for this cycle. Table 5.1 indicates that Eta12 performs better than GFS at the four key locations. The difference in rms error at station 42035, for dot product, is almost 2 m/s. Once again, the plots would seem to confirm this finding. Figures 5.2 and 5.3 indicate that during the high windspeed event of mid July, which was Hurricane Claudette, Eta12 performed much better with a maximum windspeed of 16 m/s, while it was only 11 m/s for GFS. The data points from the Eta12 forecast seem to conform much closer to the observed signal than do the GFS forecast data points. However, GFS did better at forecasting the atmospheric pressure.

Table 5.1 Wind Statistical Analysis : July 2003, 00z forecast

Rmse1 is the rms error of the model windspeed to the observed windspeed. Rmse2 is the rms error of the model windspeed in the direction of the observed wind. Rmse3 is the rms error of the atmospheric pressure.

	GFS			Eta12		
Stations	rmse1(m/s)	rmse2(m/s)	rmse3(mb)	rmse1(m/s)	rmse2(m/s)	rmse3(mb)
42035	2.969	4.967	1.910	2.527	3.113	3.609
42019	3.094	3.729	1.930	2.910	4.569	3.643
42020	3.079	3.732	1.155	2.201	2.585	3.892
42002	4.001	5.918	1.585	1.815	3.439	3.117
42001	3.308	4.371	1.448	2.534	2.813	2.622
42041	1.527	1.895	0.980	2.254	2.943	3.077
42040	3.046	4.522	1.291	2.524	3.177	1.426
42039	2.862	3.849	1.160	2.342	3.021	3.040
42036	2.340	2.990	0.836	1.870	2.659	2.880
SRST2	3.386	3.930	2.273	1.907	1.818	3.576
PTAT2	2.656	2.470	0.992	3.460	3.061	3.653

Table 5.2 Forecast Wind Evaluation, 00z forecast, July 2003

	GFS	Eta12
Station	npf	npf
42035	11	14
42020	3	22
42001	3	22
42036	12	13

Note: npf is the number of preferred forecasts based on rmse. Also, Eta12 missed six forecasts during July 2003.

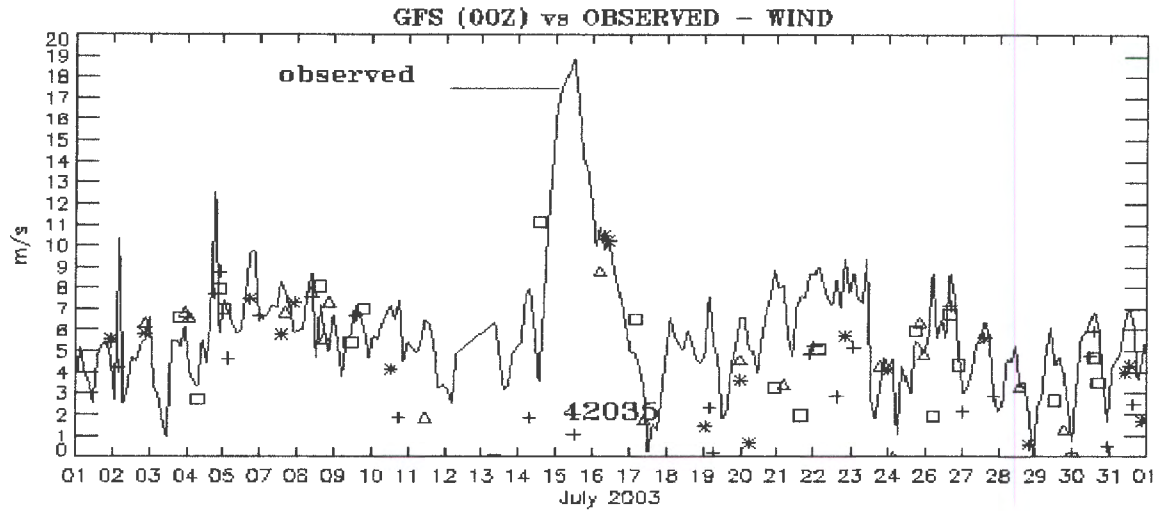


Figure 5.2 GFS (00z) vs. Observed Wind at Station 42035, July 2003

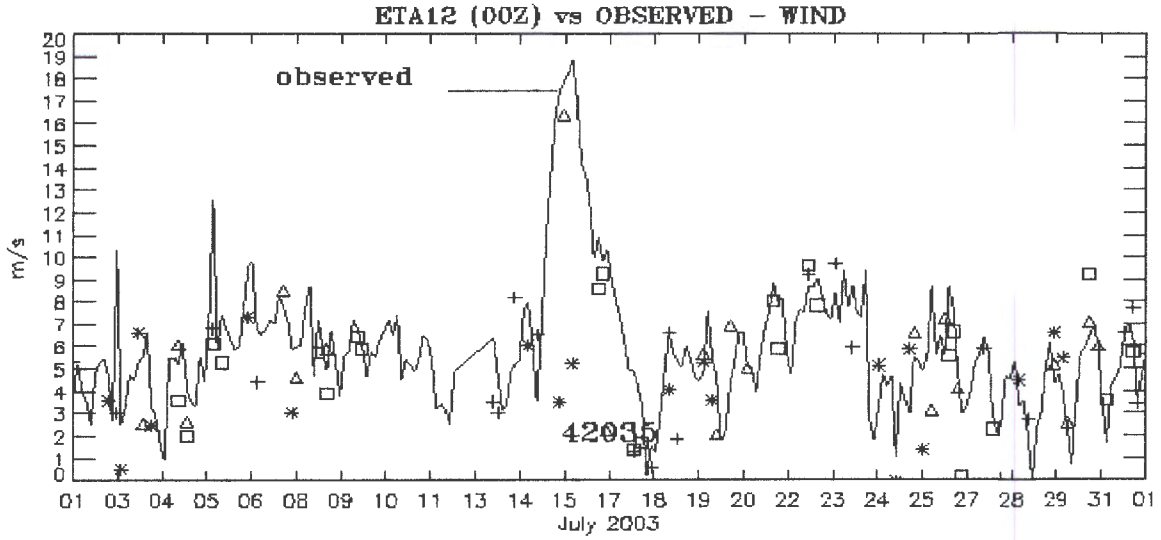


Figure 5.3 Eta12 (00z) vs. Observed Wind at Station 42035, July 2003

Note : Four points are plotted using a common symbol for each daily forecast. The four plotted points include the high, low, start, and end point. Symbols used to represent forecast these points include the plus, square, triangle, and asterisk, respectively.

6. CONCLUSIONS

Wind and sea level atmospheric pressure comparisons of observations with forecast Eta12 and GFS model output were performed for the months of November 2002, January 2003, May 2003, and July 2003. Windfield comparisons considered windspeed, windspeed in the direction of the observed wind, and atmospheric pressure at 12 stations throughout the Gulf of Mexico. In all months, Eta12 model winds compared more favorably to the observations than the coarser resolution GFS model winds at the majority of the stations. During hurricane Claudette in July 2003, the Eta12 winds were far superior to the GFS winds at station 42035 (off shore from Galveston, TX).

This report is a companion to the water level comparison report (Richardson and Schmalz, 2004). In that report, forecast subtidal water levels produced by the DGOM model, which uses US Navy COAMPS forcing, were compared to forecast subtidal water levels from the ETSS model, which uses the NWS GFS forcing. The present report allows for the analysis of wind and atmospheric pressure fields separately and prior to the analysis of the subtidal water levels. In this manner, differences in water levels may be related to differences in the wind and atmospheric pressure forcings in a quantitative manner.

Future enhancements to the wind and atmospheric pressure analysis procedure include the following:

1. In the monthly summary tables, npf is defined to be the number of preferred forecasts. If the Eta12 model has an rms error of 0.21 m/s and the GFS model has an rms error of 0.22 m/s, then Eta12 is the preferred forecast. In this instance however, the difference in rms error is negligible. An improvement would be to introduce a threshold value. A difference in rms error less than a specified threshold value, perhaps 0.5 m/s, would be considered a tie.
2. Improve the monthly forecast vs. observed plots by plotting the forecast windspeed points uniformly by day. Presently, the plot program, plot_wndanal.pro, cycles through the four symbols for each model regardless of forecast date. If one model misses a forecast, it becomes difficult to compare forecast points for a given day because the symbols no longer correspond.

Based on the results of this study, NOS is considering the installation of the DGOM model system (Patchen and Blaha, 2002) driven by the NWS Eta12 winds and atmospheric pressure fields. The system might also be further tested on storm event water level response and on its ability to improve the NOS Galveston Bay Experimental Nowcast/Forecast System (GBEFS) water levels (Schmalz and Richardson, 2002).

```
height = 14.94
 613          c                    longitude = 97.1
 614          c        station 13 - Port Arthur,
wind height = 6.096
 615          c        station 14 - Houston, wind
height = 6.096
 616          c        station 15 - Galveston,
wind height = 6.096
 617
 618
 619                write(6,1031)
 620                do k=1,nsta
 621
write(6,97)ista(k),stanam(ista(k))
 622                enddo
 623
 624
 625           1031 format(/,'Stations for
comparison (with obs data) : ')
 626
 627
****************************************
****************
 628
 629          C Loop thru i's and j's, call
subroutine disphr to
 630          C calculate distances from grid
cells to stations.
 631
 632                kexp =-2
 633
 634                do 50 k=1,nsta
 635                  dis_min = 9999.0
 636                  do j=1,nlat
 637                  do i=1,nlon
 638
if(opt_model.eq.'aviat')then
 639                       call
disphr(alat_avn(i,j),alon_avn(i,j),
 640                *
stalat(ista(k)),stalon(ista(k)),
 641                *                    dis)
 642
if(iwrit.eq.2)write(7,1041)i,j,stanam
 643                *                    (ista(k)),dis
 644                    endif
 645
if(opt_model.eq.'Eta12')then
 646                     call
disphr(alat_Eta(i,j),alon_Eta(i,j),
 647                *
stalat(ista(k)),stalon(ista(k)),
 648                *                    dis)
 649                    endif
 650
 651                    if(dis.lt.dis_min)then
 652                      dis_min = dis
 653                      i_dmin(ista(k)) = i
 654                      j_dmin(ista(k)) = j
 655                    endif
 656                  enddo
 657                  enddo
 658
 659                if(iwrit.eq.2)then
 660
write(7,1042)stanam(ista(k)),dis_min,i_dmin
 661                *
(ista(k)),j_dmin(ista(k))
 662                endif
 663
 664             50 continue
 665
 666
 667          C Calculate weighting factor of
each grid cell
 668
 669                do 100 k=1,nsta
 670                il(ista(k)) = i_dmin(ista(k)) -
2
 671                iu(ista(k)) = i_dmin(ista(k)) +
2
 672                jl(ista(k)) = j_dmin(ista(k)) -
2
 673                ju(ista(k)) = j_dmin(ista(k)) +
2
 674                kk = 0
```

```
 675              smd = 0.0
 676              wght_ch = 0.0
 677              do j=jl(ista(k)),ju(ista(k))
 678                do i=il(ista(k)),iu(ista(k))
 679                  kk = kk + 1
 680
if(opt_model.eq.'aviat')then
 681                    call
disphr(alat_avn(i,j),alon_avn(i,j),
 682             *
stalat(ista(k)),stalon(ista(k)),
 683             *                  dis)
 684                  endif
 685
if(opt_model.eq.'Eta12')then
 686                    call
disphr(alat_Eta(i,j),alon_Eta(i,j),
 687             *
stalat(ista(k)),stalon(ista(k)),
 688             *                  dis)
 689                  endif
 690
if(iwrit.eq.3)write(7,1041)i,j,stanam(ista(k)),
 691             *                        dis
 692                  d(ista(k),kk) = dis
 693                  smd = smd +
d(ista(k),kk)**kexp
 694                enddo
 695              enddo
 696
 697              do n=1,kk
 698                wt(ista(k),n) =
d(ista(k),n)**kexp/smd
 699                wght_ch = wght_ch +
wt(ista(k),n)
 700
if(iwrit.eq.3)write(7,1043)wt(ista(k),n)
 701              enddo
 702
 703              if(iwrit.eq.3)then
 704                write(7,1044)wght_ch
 705              endif
 706
 707            100 continue
 708
 709
 710            1041 format(1x,'Distance from
cell ',i3,',',i3,
 711             *       ' to station ',a5,' is',f9.2,'
km')
 712            1042 format(1x,a5,f8.2,'
km',2i5)
 713            1043 format(1x,'The weighting
factor is ',f9.4)
 714            1044 format(1x,'sum of
weighting factors = ',f5.2,/)
 715
 716
****************************************
****************
 717
 718            c Read forecast model data -
aviation, Eta12, or EDAS
 719
 720                if(opt_model.eq.'aviat')then
 721
 722            c Read Aviation forecast data
from binary file,
 723            c perform necessary conversions
to get u and v components
 724            c of wind into m/s and air
pressure into millibars.
 725            c Check to make sure id (day
value in loop) equals
 726            c iday (day value read from
binary file).
 727
 728                  id00 = 1
 729                  do 190 id=idays,idayf
 730
 731                    if(id.ne.numd(id00))then
 732            c        do 200 n=1,8
 733                     do 200 n=1,13
 734
read(lun1(id))isec,iyear,imonth,iday,ihour
 735
```

```
 736
if(iwrit.eq.4)write(7,1054)isec,iyear,imonth,i
day,
 737          *                          ihour
 738
read(lun1(id))iu00,iv00,ip00
 739
 740                if(id.ne.iday)then
 741                  if(n.eq.1)write(6,802)
 742                  write(6,1055)n,id,iday
 743                  stop
 744                endif
 745              do 205 k=1,nsta
 746              kk = 0
 747              do j=jl(ista(k)),ju(ista(k))
 748              do i=il(ista(k)),iu(ista(k))
 749                kk = kk + 1
 750                fu00(ista(k),iday,n,kk) =
float(iu00(i,j))/10.0
 751                fv00(ista(k),iday,n,kk) =
float(iv00(i,j))/10.0
 752                fp00(ista(k),iday,n,kk) =
(float(ip00(i,j))+10000.0)
 753          *                          *10.0
 754
 755
if(i.eq.il(ista(k)).and.j.eq.jl(ista(k)))then
 756                  if(iwrit.eq.4)then
 757
write(7,1052)iday,ihour
 758                    write(7,1053)
 759                  endif
 760                endif
 761
 762                if(iwrit.eq.4)then
 763
write(7,1051)i,j,fu00(ista(k),id,n,kk),
 764          *
fv00(ista(k),id,n,kk),fp00(ista(k),id,n,kk)
 765
write(7,*)alat_avn(i,j),alon_avn(i,j)
 766                endif
 767              enddo
 768              enddo
 769
 770          205   continue
 771          200   continue
 772              else
 773                id00 = id00 + 1
 774              endif
 775
 776          190   continue
 777            endif
 778
 779
 780          1051
format(1x,2i3,2f10.2,f13.2)
 781          1052 format(1x,'Day ',i2,' Hour
',i2)
 782          1053 format(1x,' I  J
UCOMP     VCOMP    PRESSURE')
 783          1054 format(1x,i8,4i5)
 784          1055 format('n = ',i2,' id = ',i3,
 785            *      ', day read from 12z bin
file =',i3)
 786
 787
!-------------------------------------------------
 788
 789          c Call subroutine readuvp to read
Eta netcdf file for
 790          c U and V components of wind,
and air pressure.  Loops
 791          c 211, 212, 213 map u
component, v component, and pressure
 792          c values from netcdf structure
arrays to our own desired
 793          c structure arrays.
 794
 795            if(opt_model.eq.'Eta12')then
 796              id00 = 1
 797
 798              do 225 idy=idays,idayf
 799
 800              if(idy.ne.numd(id00))then
 801                call
```

```
readuvp(iwrit,filenmEta(idy),ndim,rjday,
     *
nlon,nlat,unetcdf,vnetcdf,pnetcdf)

                do 211 n=1,ndim
                  do 212 j=1,nlat
                    do 213 i=1,nlon_Eta
                      u(n,i,j) =
unetcdf(j,i,n)
                      v(n,i,j) =
vnetcdf(j,i,n)
                      p(n,i,j) =
pnetcdf(j,i,n)
          213         continue
          212       continue
          211     continue

                if(iwrit.eq.14)then

write(6,1064)unetcdf(1,1,1),vnetcdf(1,1,1),
          *                 pnetcdf(1,1,1)

write(6,1064)u(1,1,1),v(1,1,1),p(1,1,1)

write(6,1064)unetcdf(nlat_Eta,1,1),vnetcdf
          *
(nlat_Eta,1,1),pnetcdf(nlat_Eta,1,1)

write(6,1064)u(1,1,nlat),v(1,1,nlat),
          *                 p(1,1,nlat)
                endif

                nframs = 13


if(iwrit.eq.4)write(7,1062)
                do 270 n=3,nframs
                do 279 nk=1,nsta
                  kk = 0
                  doj=jl(nk),ju(nk)
                    do i=il(nk),iu(nk)
                    kk = kk + 1
                    if(iwrit.eq.4)then

write(7,1061)stanam(nk),idy,n,kk,u(n,i,j),
          *
v(n,i,j),p(n,i,j)
                    endif
                    fu(nk,idy,n,kk) =
u(n,i,j)
                    fv(nk,idy,n,kk) =
v(n,i,j)
                    fp(nk,idy,n,kk) =
p(n,i,j)
                    enddo
                  enddo
          279     continue
          270     continue

              if(iwrit.eq.4)write(7,1062)

              else
                id00 = id00 + 1
              endif

          225   continue
            endif


          1061
format(2x,a5,3i6,2f10.2,f12.2)
          1062 format(/,1x,'Station   day
nhr  kk    U vel  ',
          *         ' V vel   Pressure')
          1064 format(1x,3f12.4)


**********************************************
***************
c     Calculate wind height
adjustment factor for each station

              do 215 jh=1,nsta

if(iwrit.eq.7)write(11,82)ista(jh),wind_height
```

```
(ista(jh))
 866
if(wind_height(ista(jh)).le.20.0)then
 867                  u_ten(ista(jh)) =
(10.0/wind_height(ista(jh)))**
 868             *                  (1.0/7.0)
 869                else
 870                  u_ten(ista(jh)) =
log(10.0/.01)/log(wind_height
 871             *            (ista(jh)) /0.01)
 872                endif
 873
if(iwrit.eq.7)write(11,83)ista(jh),u_ten(ista(jh
))
 874          215 continue
 875
 876
 877           82 format(1x,'Wind Height for
station ',i2,' is',f6.2)
 878           83 format(1x,'Wind adjustment
for station ',i2,' = ',
 879             *     f8.5)
 880
 881
**************************************
***************
 882
 883          c    Read observed data
 884          c    Observed wind directions
are given in meteorological
 885          c    convention (direction from
which it is blowing).
 886          c    Buoy and C-Man data are in
Grenwich time.  Stations
 887          c    4 (Port Arthur), 5 (Houston),
and 6 (Galveston, for 1997)
 888          c    are read from airport data
which is in local time.
 889
 890
!-----------------------------------------------------
 891
 892          c    For itransopt gt 0, open
observed transfer plot
 893          c    files.
 894
 895          c    obsplot - filename for
observed transfer plot file
 896
 897               if(itransopt.gt.0)then
 898                 do k=1,nsta
 899                   read(5,498)obsplot(k)
 900                 enddo
 901
 902                 lunob_trans = 83
 903                 do k=1,nsta
 904                   lunobtr(k) = lunob_trans
 905
 906
open(lunobtr(k),file=obsplot(k),form='format
ted')
 907
 908                   lunob_trans =
lunob_trans + 1
 909                 enddo
 910               endif
 911
 912
 913           498 format(1x,a13)
 914
 915
!-----------------------------------------------------
 916
 917          c    Read header line from each
obs file.
 918               do k=1,nsta
 919                 icnt0(ista(k)) = 0
 920                 read(lun(ista(k)),1015)line
 921               enddo
 922
 923
 924               if(iwrit.eq.8)write(9,521)
 925               do 500 k=1,nsta
 926                 id00 = 1
 927                 nskip(ista(k)) = 0
 928
```

```
 929
if(iwrit.eq.5)write(11,520)ista(k),stanam(ista(
k))
 930
if(iwrit.eq.17)write(6,520)ista(k),stanam(ista(
k))
 931
if(iwrit.eq.18)write(82,520)ista(k),stanam(ist
a(k))
 932
 933                idcnt = 1
 934
 935                rjstrt = forcstrt
 936                rjstop = forcstop
 937
 938          501   continue
 939
read(lun(ista(k)),502,end=551)myr,imon,iday
,ihr,
 940           *
iwd,ws,press,atemp,wetb_temp
 941
 942
if(idcnt.eq.numd(id00))then
 943                  if(iwrit.eq.17)then
 944
write(6,526)stanam(ista(k)),imon,idcnt,iday,
 945           *                  ihr
 946                endif
 947                if(iday.eq.idcnt+1)then
 948                  if(ihr.eq.12)then
 949                    if(iwrit.eq.17)then
 950
write(6,*)id00,numd(id00)
 951                      write(6,535)idcnt
 952                    endif
 953                    idcnt = idcnt + 1
 954                    do l=1,9
 955
backspace(lun(ista(k)))
 956                    enddo
 957                    rjstrt = rjstrt + 1.0
 958                    rjstop = rjstop + 1.0
 959                    id00 = id00 + 1
 960                  endif
 961                endif
 962                goto 501
 963              endif
 964
 965              myrwr = myr - 2000
 966
if(iwrit.eq.17)write(6,522)imon,iday,myrwr,i
hr
 967              call
calcjd(julday,imon,iday,myr)
 968              rjulday = julday
 969
if(iwrit.eq.17)write(6,*)rjulday
 970              rjulday = rjulday +
float(ihr)/24.0
 971
if(iwrit.eq.17)write(6,*)rjulday
 972
 973              if(rjulday.ge.rjstrt)then
 974                if(iday.eq.idcnt)then
 975                  nhr = ihr/3 + 1
 976                endif
 977                if(iday.gt.idcnt)then
 978                  nhr = nhr + 1
 979                  if(iwrit.eq.17)then
 980
write(6,*)idcnt,iday,nhr
 981                  endif
 982                endif
 983                if(iwrit.eq.17)then
 984                  write(6,*)'nhr ',nhr,' ihr
',ihr
 985                endif
 986              else
 987                goto 501
 988              endif
 989
 990
if(iwrit.eq.5)write(11,523)imon,iday,ihr,ws,i
wd,press
 991
```

```
 992
!-----------------------------------------------
 993
 994          c       Check for null values
 995
 996          c     variables :
 997          c       nskip() - counter for
number of null (obs) values
 998          c      imskip() - month of null
value (imon)
 999          c      idskip() - day of null value
(iday)
 1000         c     icntskip() - hour of null value
(nhr)
 1001
 1002
if(press.eq.9999.0.or.ws.gt.98.9)then
 1003                    if(iwrit.eq.11)then
 1004
write(12,527)stanam(ista(k)),imon,iday,ihr,
 1005             *                    ws,iwd,press
 1006                    endif
 1007                    if(iwrit.eq.5)then
 1008
write(11,528)stanam(ista(k)),imon,iday,ihr,
 1009             *                    ws,iwd,press
 1010                    endif
 1011                    if(iday.ge.idays)then
 1012                      nskip(ista(k)) =
nskip(ista(k)) + 1
 1013
imskip(nskip(ista(k)),ista(k)) = imon
 1014
idskip(nskip(ista(k)),ista(k)) = idcnt
 1015
icntskip(nskip(ista(k)),ista(k)) = nhr
 1016                    if(iwrit.eq.11)then
 1017
write(12,533)ista(k),stanam(ista(k)),
 1018             *                    imon,iday,ihr
 1019
write(12,534)imon,idcnt,nhr
 1020                    endif
 1021                  endif
 1022                endif
 1023
 1024
 1025          527
format(2x,a5,1x,3i3,f6.1,i4,f8.2)
 1026          528
format(2x,a5,1x,3i3,f6.1,i4,f8.2,/)
 1027          533 format(1x,'Station
',i3,',1x,a5,' actual time - ',
 1028             *       i2,'/',i2,' hour',i3)
 1029          534 format(11x,'Forecast period
time - ',i2,'/',i2,
 1030             *         ' nhr =',i2)
 1031
 1032
!-----------------------------------------------
 1033
 1034                ws_obs(idcnt,nhr,ista(k)) =
ws
 1035
 1036                if(iwrit.eq.17)then
 1037
write(6,531)iday,nhr,ista(k),ws_obs(idcnt,nhr
,
 1038             *                    ista(k))
 1039                endif
 1040                if(iwrit.eq.18)then
 1041
write(82,531)iday,nhr,ista(k),ws_obs(idcnt,n
hr,
 1042             *                    ista(k))
 1043                endif
 1044          c       Wind speeds from buoy
(42035) are read in m/s
 1045          c       (iw=2).  Wind speeds from
C-man stations are read
 1046          c       in knots (iw=0).
 1047                if(iw(ista(k)).eq.0)then
 1048
if(ws.lt.98.8.and.press.ne.9999.0)then
 1049
ws_obs(idcnt,nhr,ista(k)) =
```

```
ws_obs(idcnt,nhr,
 1050           *           ista(k)) *
conv_ktsms
 1051
if(iwrit.eq.5)write(11,525)ws,ws_obs(idcnt,
 1052           *                nhr,ista(k))
 1053
ws_obs10(idcnt,nhr,ista(k)) =
ws_obs(idcnt,nhr,
 1054           *           ista(k)) *
u_ten(ista(k))
 1055                 if(iwrit.eq.17)then
 1056
write(6,531)iday,nhr,ista(k),ws_obs10
 1057           *
(idcnt,nhr,ista(k))
 1058                 endif
 1059                 if(iwrit.eq.18)then
 1060
write(82,531)iday,nhr,ista(k),ws_obs10
 1061           *
(idcnt,nhr,ista(k))
 1062                 endif
 1063
if(iwrit.eq.5)write(11,524)ws_obs(idcnt,nhr,
 1064           *
ista(k)),ws_obs10(idcnt,nhr,ista(k))
 1065                 else
 1066
ws_obs10(idcnt,nhr,ista(k)) = 99999999.0
 1067                 endif
```

Program Listing 2.1 Forc.aviEta.f (continued)

```
 1068               endif
 1069
 1070               if(iw(ista(k)).eq.2)then
 1071                 if(ws.lt.98.8)then
 1072
ws_obs10(idcnt,nhr,ista(k)) = ws_obs(idcnt,
 1073           *           nhr,ista(k)) *
u_ten(ista(k))
 1074                 if(iwrit.eq.17)then
 1075
write(6,531)iday,nhr,ista(k),ws_obs10
 1076           *
(idcnt,nhr,ista(k))
 1077                 endif
 1078                 if(iwrit.eq.18)then
 1079
write(82,531)iday,nhr,ista(k),ws_obs10
 1080           *
(idcnt,nhr,ista(k))
 1081                 endif
 1082
if(iwrit.eq.5)write(11,524)ws_obs(idcnt,nhr,
 1083           *
ista(k)),ws_obs10(idcnt,nhr,ista(k))
 1084                 else
 1085
ws_obs10(idcnt,nhr,ista(k)) = 99999999.0
 1086                 endif
 1087               endif
 1088
 1089
!-------------------------------------------------
 1090
 1091       c Write observed data to plot
transfer files
 1092
 1093       c      lunobtr() - logical unit
numbers for plot trans
 1094       c                files
 1095
 1096
if(itransopt.gt.0.and.nhr.le.10)then
 1097
write(lunobtr(k),369)rjulday,ws_obs10(idcnt,
nhr,
 1098           *           ista(k)),nhr
 1099             endif
 1100
 1101
!-------------------------------------------------
 1102
 1103               if(press.ne.9999.0)then
```

```
 1104
press_obs(idcnt,nhr,ista(k)) = press
 1105               else
 1106
press_obs(idcnt,nhr,ista(k)) = 99999999.0
 1107               endif
 1108
 1109               rwd = float(iwd)
 1110
 1111               call
uvcomp(ws_obs10(idcnt,nhr,ista(k)),rwd,
 1112              *
ucompobs,vcompobs,dirplt)
 1113               wdir_obs(idcnt,nhr,ista(k))
= dirplt
 1114
ucomp_obs(idcnt,nhr,ista(k)) = -ucompobs
 1115
vcomp_obs(idcnt,nhr,ista(k)) = -vcompobs
 1116
 1117               if(iwrit.eq.8)then
 1118
write(9,529)ista(k),stanam(ista(k)),myr,imon,
 1119              *
iday,ihr,iwd,ws_obs10(idcnt,nhr,ista(k)),
 1120              *
press_obs(idcnt,nhr,ista(k))
 1121               endif
 1122
 1123               do l=1,2
 1124
read(lun(ista(k)),1015)line
 1125               enddo
 1126
 1127
 1128               if(rjulday.ge.rjstop)then
 1129
if(iwrit.eq.17)write(6,532)idcnt
 1130
if(iwrit.eq.18)write(82,532)idcnt
 1131                 if(idcnt.eq.idaystop)then
 1132                   goto 551
 1133                 endif
 1134                 idcnt = idcnt + 1
 1135                 do l=1,9
 1136                   backspace(lun(ista(k)))
 1137                 enddo
 1138                 rjstrt = rjstrt + 1.0
 1139                 rjstop = rjstop + 1.0
 1140               endif
 1141
 1142               goto 501
 1143          551  continue
 1144
 1145               if(nskip(ista(k)).gt.0)then
 1146
if(iwrit.eq.11)write(12,802)
 1147               endif
 1148          500 continue
 1149
 1150
 1151          502
format(i4,3(i3),i4,f5.1,28x,3f6.1)
 1152          520 format(/,1x,'Observed Data
- Station',i2,2x,a5)
 1153          521
format(/,20x,'Observed',/,1x,' Station  Year
Mon day',
 1154              *     ' hour dir  spd(m/s)
pressure')
 1155          522 format(/,1x,i2,'/',i2,'/',i2,'
hour',i2)
 1156          523 format(1x,3i3,f6.1,i4,f8.2)
 1157          524 format(1x,f6.2,'m/s
converted to 10m height value of',
 1158              *       f6.2,'m/s',/)
 1159          525 format(1x,' Wind
speed;',f5.1,' knots converted to',
 1160              *                  f6.2,'m/s')
 1161          526 format(1x,a5,2i4,2i5)
 1162          529
format(1x,i2,2x,a5,2x,i4,i3,2(i4),i6,f8.2,f9.1)
 1163          531 format(1x,3i3,f9.4)
 1164          532 format(1x,'End of daily
forecast, day ',i3)
 1165          535 format(1x,'End of daily
```

```
forecast, day ',i3,/,
 1166          *      1x,'day of missing
forecast file')
 1167
 1168
**************************************
***************
 1169
 1170       c    write observed data points to
be skipped
 1171
 1172            if(iwrit.eq.11)then
 1173              write(12,541)
 1174              do k=1,nsta
 1175                do n=1,nskip(ista(k))
 1176
write(12,542)stanam(ista(k)),imskip(n,ista(k)
),
 1177          *
idskip(n,ista(k)),icntskip(n,ista(k))
 1178                enddo
 1179              enddo
 1180              write(12,802)
 1181            endif
 1182
 1183
 1184       c    Initialize monthly total files
(*_obs_tot).
 1185       c    Store observed values of U
component, V component, wind
 1186       c    speed, air pressure in
cumulative arrays (ucomp_obs_cum,
 1187       c    vcomp_obs_cum,
ws_obs_cum, and press_obs_cum).
 1188
 1189
 1190            do k=1,nsta
 1191              npt_obs(ista(k)) = 0
 1192              ws_obs10_tot(ista(k)) = 0.0
 1193              ucomp_obs_tot(ista(k)) =
0.0
 1194              vcomp_obs_tot(ista(k)) =
0.0
 1195              press_obs_tot(ista(k)) = 0.0
 1196              nskip(ista(k)) = 1
 1197            enddo
 1198
 1199            if(iwrit.eq.11)then
 1200              write(12,562)
 1201              write(12,563)
 1202            endif
 1203
 1204            id00 = 1
 1205
 1206
 1207            do 545 id=idays,idayf
 1208              if(id.ne.nurnd(id00))then
 1209              if(id.eq.idayf)then
 1210                iendhr = 8
 1211              else
 1212                iendhr = 13
 1213              endif
 1214              do 548 ih=3,iendhr
 1215                do 550 k=1,nsta
 1216
if(id.eq.idskip(nskip(ista(k)),ista(k)).and.ih
 1217          *
.eq.icntskip(nskip(ista(k)),ista(k)))then
 1218
if(iwrit.eq.11)write(12,564)stanam(ista(k)),
 1219          *
id,idskip(nskip(ista(k)),ista(k)),ih,
 1220          *
icntskip(nskip(ista(k)),ista(k))
 1221                      npt_obs(ista(k)) =
npt_obs(ista(k))
 1222                      nskip(ista(k)) =
nskip(ista(k)) + 1
 1223                    else
 1224                      npt_obs(ista(k)) =
npt_obs(ista(k)) + 1
 1225
 1226
ws_obs_cum(ista(k),npt_obs(ista(k)))
 1227          *                   =
ws_obs10(id,ih,ista(k))
```

```
 1228
if(ws_obs10(id,ih,ista(k)).gt.70000.0)then
 1229
write(6,2001)id,ih,ws_obs10(id,ih,ista(k))
 1230                        write(6,*)'Program
stopped'
 1231                        stop
 1232                       endif
 1233                       ws_obs10_tot(ista(k))
= ws_obs10_tot(ista(k))
 1234            *                  +
ws_obs10(id,ih,ista(k))
 1235
 1236
ucomp_obs_cum(ista(k),npt_obs(ista(k)))
 1237            *                  =
ucomp_obs(id,ih,ista(k))
 1238
ucomp_obs_tot(ista(k)) =
ucomp_obs_tot(ista(k))
 1239            *                  +
ucomp_obs(id,ih,ista(k))
 1240
 1241
vcomp_obs_cum(ista(k),npt_obs(ista(k)))
 1242            *                  =
vcomp_obs(id,ih,ista(k))
 1243
vcomp_obs_tot(ista(k)) =
vcomp_obs_tot(ista(k))
 1244            *                  +
vcomp_obs(id,ih,ista(k))
 1245
 1246
press_obs_cum(ista(k),npt_obs(ista(k)))
 1247            *                  =
press_obs(id,ih,ista(k))
 1248                       press_obs_tot(ista(k))
= press_obs_tot(ista(k))
 1249            *                  +
press_obs(id,ih,ista(k))
 1250                      endif
 1251          550         continue
 1252          548    continue
 1253                 else
 1254                   id00 = id00 + 1
 1255                 endif
 1256          545 continue
 1257
 1258
 1259          541 format(/,'Data Points to be
skipped')
 1260          542 format(1x,a5,2x,i2,'/',i2,',
nhr=',i3)
 1261          562 format(' Store observed
values in cumulative Arrays')
 1262         2001 format(1x,2i4,f12.1)
 1263
 1264
!-------------------------------------------------
 1265
 1266          c    Calculate mean wind speed,
U component, V component, and
 1267          c    air pressure.  Call subroutine
sigma to calculate standard
 1268          c    deviation for each ndat.
Write station name, number of data
 1269          c    points, and mean wind
speed.  ws_avg_obs(ista(k)) is mean
 1270          c    observed wind speed.
 1271
 1272
 1273              data_type = 'obs'
 1274              write(6,571)
 1275              if(iwrit.eq.13)write(13,579)
 1276              do k=1,nsta
 1277
if(iwrit.eq.13)write(13,382)stanam(ista(k))
 1278                 ws_avg_obs(ista(k)) =
ws_obs10_tot(ista(k))/
 1279            *          npt_obs(ista(k))
 1280                 call
sigma(ista(k),1,iwrit,data_type,standev)
 1281
if(iwrit.eq.13)write(13,581)standev
 1282                 rmonth_stat(ista(k),3,1) =
```

```
standev
 1283
 1284                ucomp_avg_obs(ista(k)) =
ucomp_obs_tot(ista(k))/
 1285           *
npt_obs(ista(k))
 1286                call
sigma(ista(k),2,iwrit,data_type,standev)
 1287
if(iwrit.eq.13)write(13,582)standev
 1288                rmonth_stat(ista(k),3,2) =
standev
 1289
 1290                vcomp_avg_obs(ista(k)) =
vcomp_obs_tot(ista(k))/
 1291           *
npt_obs(ista(k))
 1292                call
sigma(ista(k),3,iwrit,data_type,standev)
 1293
if(iwrit.eq.13)write(13,583)standev
 1294                rmonth_stat(ista(k),3,3) =
standev
 1295
 1296                press_avg_obs(ista(k)) =
press_obs_tot(ista(k))/
 1297           *
npt_obs(ista(k))
 1298                call
sigma(ista(k),4,iwrit,data_type,standev)
 1299
if(iwrit.eq.13)write(13,584)standev
 1300                rmonth_stat(ista(k),3,4) =
standev
 1301
 1302
if(npt_obs(ista(k)).eq.0)ws_avg_obs(ista(k))
= 999999.9
 1303                rmonth_stat(ista(k),1,1) =
ws_avg_obs(ista(k))
 1304                rmonth_stat(ista(k),1,2) =
ucomp_avg_obs(ista(k))
 1305                rmonth_stat(ista(k),1,3) =
vcomp_avg_obs(ista(k))
 1306                rmonth_stat(ista(k),1,4) =
press_avg_obs(ista(k))
 1307
write(6,1501)stanam(ista(k)),npt_obs(ista(k))
,
 1308           *
ws_obs10_tot(ista(k)),ws_avg_obs(ista(k))
 1309           enddo
 1310
 1311
 1312          561 format(/,1x,'Cumulative
Observed data')
 1313          571 format(/,'Number of
observed points by station',/,
 1314           *       'station    number
total     mean')
 1315          579 format(/,' File for monthly
statistics')
 1316          581 format(' standard deviation
obs wind speed = ',f7.3)
 1317          582 format(' standard deviation
obs U comp = ',f7.3)
 1318          583 format(' standard deviation
obs V comp = ',f7.3)
 1319          584 format(' standard deviation
obs air press = ',f7.3)
 1320
 1321
**************************************
****************
 1322
 1323
!---------------------------------------------------
 1324
 1325          c Set wind height correction
factor for model data.
 1326
 1327          c     u_ten_nmc - conversion
factor for model data to 10m.
 1328
 1329          c
if(opt_model.eq.'aviat'.and.avi_opt.eq.'regula
```

```
r')then
 1330         c      u_ten_nmc =
log(10.0/0.01)/log(35/.01)
 1331         c    else
 1332                u_ten_nmc = 1.0
 1333         c    endif
 1334
 1335              if(iwrit.eq.7)write(11,*)'
u_ten_nmc = ',u_ten_nmc
 1336              write(80,371)u_ten_nmc
 1337
 1338
 1339           371 format(/,'AVN height
adjustment to 10m = ',f6.3)
 1340
 1341
!-----------------------------------------------------
 1342
 1343         c Initialize array values :
 1344         c
 1345         c     npt_mod - number of model
values
 1346         c ws_mod_tot() -
 1347
 1348              do k=1,nsta
 1349                npt_mod(ista(k)) = 0
 1350                nskip(ista(k)) = 1
 1351                ws_mod_tot(ista(k)) = 0.0
 1352                ucomp_mod_tot(ista(k)) =
0.0
 1353                vcomp_mod_tot(ista(k)) =
0.0
 1354              enddo
 1355
 1356              if(iwrit.eq.8)write(9,372)
 1357
 1358
!-----------------------------------------------------
 1359
 1360         c    Initialize luntrans(k), logical
unit numbers for
 1361         c    plot transfer files (model
data).
 1362         c
 1363         c    luntrans() - logical unit
number (model data) for
 1364         c              plot transfer files.
 1365
 1366              if(itransopt.gt.0)then
 1367                luntran = 95
 1368                do k=1,nsta
 1369                read(5,392)fileplot(k)
 1370                luntrans(k) = luntran
 1371
 1372                luntran = luntran + 1
 1373                enddo
 1374              endif
 1375
 1376
 1377           392 format(1x,a12)
 1378
 1379
!-----------------------------------------------------
 1380
 1381         c Processing of model data.
 1382         c Use weighting factors to
calculate contribution
 1383         c (U component, V component,
and air pressure)
 1384         c from each grid cell to the 6
station locations.
 1385
 1386              if(iwrit.eq.11)then
 1387                write(12,391)
 1388                write(12,563)
 1389              endif
 1390
 1391              id00 = 1
 1392
 1393              do 390 iday=idays,idayf
 1394
if(iwrit.eq.6)write(7,381)iday
 1395
 1396
 1397                if(iday.ne.numd(id00))then
 1398                if(iday.eq.idayf)then
```

```
 1399                 iendhr = 8
 1400               else
 1401                 iendhr = 13
 1402               endif
 1403
 1404         c    All 8 values for each day,
hours 0, 3, 6, 9, 12,
 1405         c    15, 18, 21 are taken from .00
file.
 1406
 1407               do 400 nhr=3,iendhr
 1408                 ihr = (nhr - 1) * 3
 1409
 1410                 lunout = 7
 1411
 1412                 do 300 k=1,nsta
 1413                   kk = 0
 1414                   ucom_tot = 0.0
 1415                   vcom_tot = 0.0
 1416                   press_tot = 0.0
 1417
 1418
if(iwrit.eq.6)write(lunout,382)stanam(ista(k))
 1419
 1420                   if(iday.le.9)then
 1421                     write(cday1,'(i1)')iday
 1422                     cday = '0'//cday1
 1423                   else
 1424                     write(cday,'(i2)')iday
 1425                   endif
 1426
 1427                   fname =
fileplot(k)//cday
 1428                   call ncrght(fname,nchr)
 1429
 1430                   if(itransopt.gt.0)then
 1431
open(luntrans(k),file=fname(1:nchr),
 1432              *
form='formatted')
 1433                   endif
 1434
 1435                   do
j=jl(ista(k)),ju(ista(k))
 1436                 do
i=il(ista(k)),iu(ista(k))
```

Program Listing 2.1 Forc.aviEta.f (continued)

```
 1437                   kk = kk + 1
 1438
if(opt_model.eq.'aviat')then
 1439
if(iday.ne.numd(id00))then
 1440                     ucom_staloc =
wt(ista(k),kk)*fu00
 1441              *
(ista(k),iday,nhr,kk)
 1442                     vcom_staloc =
wt(ista(k),kk)*fv00
 1443              *
(ista(k),iday,nhr,kk)
 1444                     press_staloc =
wt(ista(k),kk)*fp00
 1445              *
(ista(k),iday,nhr,kk)
 1446                   endif
 1447                   if(iwrit.eq.6)then
 1448
write(7,379)nhr,i,j,fu00(ista(k),iday,
 1449              *
nhr,kk),wt(ista(k),kk),ucom_staloc
 1450
write(7,379)nhr,i,j,fv00(ista(k),iday,
 1451              *
nhr,kk),wt(ista(k),kk),vcom_staloc
 1452
write(7,379)nhr,i,j,fp00(ista(k),iday,
 1453              *
nhr,kk),wt(ista(k),kk),press_staloc
 1454                   endif
 1455                   endif
 1456
 1457
if(opt_model.eq.'Eta12')then
 1458                     ucom_staloc =
```

```
wt(ista(k),kk) * fu(ista(k),
 1459            *
iday,nhr,kk)
 1460                        vcom_staloc =
wt(ista(k),kk) * fv(ista(k),
 1461            *
iday,nhr,kk)
 1462                        press_staloc =
wt(ista(k),kk) * fp
 1463            *
(ista(k),iday,nhr,kk)
 1464                        if(iwrit.eq.6)then
 1465
write(7,379)nhr,i,j,fu(ista(k),iday,nhr,
 1466            *
kk),wt(ista(k),kk),ucom_staloc
 1467
write(7,379)nhr,i,j,fv(ista(k),iday,nhr,
 1468            *
kk),wt(ista(k),kk),vcom_staloc
 1469
write(7,379)nhr,i,j,fp(ista(k),iday,nhr,
 1470            *
kk),wt(ista(k),kk),press_staloc
 1471                          endif
 1472                       endif
 1473
 1474                       ucom_tot =
ucom_tot + ucom_staloc
 1475                       vcom_tot =
vcom_tot + vcom_staloc
 1476                       press_tot =
press_tot + press_staloc
 1477
 1478                     enddo
 1479                   enddo
 1480
 1481
 1482                   ucom_tot_ten =
ucom_tot * u_ten_nmc
 1483                   vcom_tot_ten =
vcom_tot * u_ten_nmc
 1484
ucomp_mod(iday,nhr,ista(k)) =
ucom_tot_ten
 1485
vcomp_mod(iday,nhr,ista(k)) = vcom_tot_ten
 1486
 1487                   if(iwrit.eq.6)then
 1488
write(lunout,383)ucom_tot,vcom_tot
 1489
write(lunout,386)press_tot
 1490                   endif
 1491
 1492
 1493         c          Convert to mbars
 1494                   press_tot =
press_tot/100.0
 1495
press_mod(iday,nhr,ista(k)) = press_tot
 1496
 1497
!-------------------------------------------------------
 1498
 1499         c    Calculate resultant wind
(speed) from U and V compon-
 1500         c ents.  If idotprod equals 1, call
subroutine dotprod in
 1501         c in order to calculate the speed
of the model (Eta or GFS)
 1502         c in the direction of the observed
wind.
 1503
 1504                 if(idotopt.eq.1)then
 1505                   call
dotprod(ucomp_obs(iday,nhr,ista(k)),
 1506            *
ucomp_mod(iday,nhr,ista(k)),
 1507            *
vcomp_obs(iday,nhr,ista(k)),
 1508            *
vcomp_mod(iday,nhr,ista(k)),wsdot_model)
 1509                   wind_reslt =
wsdot_model
 1510                   if(iwrit.eq.16)then
```

```
 1511
write(6,76)ista(k),stanam(ista(k)),iday,
 1512            *
nhr,wsdot_model
 1513                endif
 1514              endif
 1515              if(idotopt.eq.0)then
 1516                wind_reslt =
sqrt(ucom_tot_ten**2 +
 1517            *
vcom_tot_ten**2)
 1518              endif
 1519
 1520
 1521         76 format(' Station ',i2,',
',a5,2i3,f9.3)
 1522
 1523
!-------------------------------------------------
 1524
 1525       c    Option for writing to plot
transfer files.
 1526
 1527              if(itransopt.gt.0)then
 1528                imon = 11
 1529                call
calcjd(julday,imon,iday,myr)
 1530                rjulday = float(julday) +
float(ihr)/24.0
 1531                time = rjulday
 1532
 1533
write(luntrans(k),369)time,wind_reslt
 1534              endif
 1535
 1536
 1537         369 format(2(1x,f9.4),i4)
 1538
 1539
!-------------------------------------------------
 1540
 1541
if(iwrit.eq.6)write(lunout,384)wind_reslt
 1542              ws_mod(iday,nhr,ista(k))
= wind_reslt
 1543
 1544              if(iwrit.eq.8)then
 1545
write(9,393)iday,ihr,ista(k),stanam(ista(k)),
 1546            *
wind_reslt,press_tot
 1547              endif
 1548
 1549
 1550       c      Store interpolated
forecast windspeed values,
 1551       c      U and V component
values, and pressure values
 1552       c      in cumulative arrays.
 1553
 1554
if(iday.eq.idskip(nskip(ista(k)),ista(k)).and.n
hr
 1555            *
.eq.icntskip(nskip(ista(k)),ista(k)))then
 1556                if(iwrit.eq.11)then
 1557
write(12,564)stanam(ista(k)),iday,idskip(nski
p
 1558            *
(ista(k)),ista(k)),nhr,icntskip(nskip
 1559            *          (ista(k)),ista(k))
 1560              endif
 1561              npt_mod(ista(k)) =
npt_mod(ista(k))
 1562              nskip(ista(k)) =
nskip(ista(k)) + 1
 1563            else
 1564              npt_mod(ista(k)) =
npt_mod(ista(k)) + 1
 1565
ws_mod_cum(ista(k),npt_mod(ista(k))) =
 1566            *                  wind_reslt
 1567              ws_mod_tot(ista(k)) =
ws_mod_tot(ista(k)) +
 1568            *
```

```
wind_reslt
 1569
 1570
ucomp_mod_cum(ista(k),npt_mod(ista(k)))
 1571           *              = ucom_tot_ten
 1572               ucomp_mod_tot(ista(k))
= ucomp_mod_tot(ista(k)) +
 1573           *
ucom_tot_ten
 1574
 1575
vcomp_mod_cum(ista(k),npt_mod(ista(k)))
 1576           *              = vcom_tot_ten
 1577               vcomp_mod_tot(ista(k))
= vcomp_mod_tot(ista(k)) +
 1578           *
vcom_tot_ten
 1579
 1580
press_mod_cum(ista(k),npt_mod(ista(k)))
 1581           *              = press_tot
 1582               press_mod_tot(ista(k)) =
press_mod_tot(ista(k)) +
 1583           *                        press_tot
 1584               endif
 1585
 1586       300     continue
 1587
 1588       400   continue
 1589             else
 1590               id00 = id00 + 1
 1591             endif
 1592       390 continue
 1593
 1594
 1595       372
format(/,20x,'Aviation',/,1x,'Day Hour
Station  Speed',
 1596           *     ' Pressure')
 1597       379 format(1x,'Hour',i3,'
cell',i3,',',i3,f10.2,f8.3,f9.3)
 1598
 1599
!-------------------------------------------------
 1600
 1601       c    Write out number of model
points by station, mean
 1602       c    wind speeds (by station).
ws_avg_mod(ista(k)) is mean
 1603       c    model speed.  Call
subroutine sigma to calculate
 1604       c    standard deviation.  Store
values for mean and standard
 1605       c    deviation in rmonth_stat.
 1606
 1607
 1608           data_type = 'mod'
 1609           write(6,398)
 1610           if(iwrit.eq.13)write(13,802)
 1611           do 305 k=1,nsta
 1612
if(iwrit.eq.13)write(13,382)stanam(ista(k))
 1613
 1614               ws_avg_mod(ista(k)) =
ws_mod_tot(ista(k))/npt_mod
 1615           *                   (ista(k))
 1616               call
sigma(ista(k),1,iwrit,data_type,standev)
 1617
if(iwrit.eq.13)write(13,681)standev
 1618               rmonth_stat(ista(k),4,1) =
standev
 1619
 1620               ucomp_avg_mod(ista(k)) =
ucomp_mod_tot(ista(k))/
 1621           *
npt_mod(ista(k))
 1622               call
sigma(ista(k),2,iwrit,data_type,standev)
 1623
if(iwrit.eq.13)write(13,682)standev
 1624               rmonth_stat(ista(k),4,2) =
standev
 1625
 1626               vcomp_avg_mod(ista(k)) =
vcomp_mod_tot(ista(k))/
```

```
 1627            *
npt_mod(ista(k))
 1628                call
sigma(ista(k),3,iwrit,data_type,standev)
 1629
if(iwrit.eq.13)write(13,683)standev
 1630                rmonth_stat(ista(k),4,3) =
standev
 1631
 1632                press_avg_mod(ista(k)) =
press_mod_tot(ista(k))/
 1633            *
npt_mod(ista(k))
 1634                call
sigma(ista(k),4,iwrit,data_type,standev)
 1635
if(iwrit.eq.13)write(13,684)standev
 1636                rmonth_stat(ista(k),4,4) =
standev
 1637
 1638
if(npt_mod(ista(k)).eq.0)ws_avg_mod(ista(k)
) = 999999.9
 1639
 1640                rmonth_stat(ista(k),2,1) =
ws_avg_mod(ista(k))
 1641                rmonth_stat(ista(k),2,2) =
ucomp_avg_mod(ista(k))
 1642                rmonth_stat(ista(k),2,3) =
vcomp_avg_mod(ista(k))
 1643                rmonth_stat(ista(k),2,4) =
press_avg_mod(ista(k))
 1644
 1645
write(6,1501)stanam(ista(k)),npt_mod(ista(k)
),
 1646            *
ws_mod_tot(ista(k)),ws_avg_mod(ista(k))
 1647          305 continue
 1648
 1649
 1650          381 format(//,'Day ',i3)
 1651          382 format(/,1x,'Station ',a5)
 1652          383 format(1x,'U Component =
',f7.3,'m/s',' V Component = ',
 1653              *       f7.3,'m/s')
 1654          384 format(1x,'Wind Speed is
',f7.3,'m/s')
 1655          386 format(1x,'Pressure is
',f10.3)
 1656          391 format(/,' Store forecast
model values in cumulative Arrays')
 1657          393
format(1x,i2,2i4,1x,a5,f7.2,f10.2)
 1658          398 format(/,'Number of model
points by station',/,
 1659              *       'station   number
total     mean')
 1660          681 format(' standard deviation
model wind speed = ',f8.3)
 1661          682 format(' standard deviation
model U comp = ',f8.3)
 1662          683 format(' standard deviation
model V comp = ',f8.3)
 1663          684 format(' standard deviation
model air press = ',f8.3)
 1664
 1665
**************************************
****************
 1666
 1667        c     Write to comparison file
(dgom.comp.0012, unit 9)
 1668        c     Write header information.
 1669
 1670              write(9,77)
 1671
if(opt_model.eq.'aviat')write(9,78)
 1672
if(opt_model.eq.'Eta12')write(9,79)
 1673
 1674              iday_old = idays
 1675              id00 = 1
 1676
 1677
 1678              do 590 iday=idays,idayf
```

```
 1679              if(iday.ne.numd(id00))then
 1680
 1681        c      Check for day increase.
formfd is to skip to next
 1682        c      page.
 1683              if(iday.gt.iday_old)then
 1684                write(9,*)formfd
 1685                write(9,77)
 1686
if(opt_model.eq.'aviat')write(9,78)
 1687
if(opt_model.eq.'Eta12')write(9,79)
 1688              endif
 1689
 1690
 1691              if(iday.eq.idayf)then
 1692                iendhr = 8
 1693              else
 1694                iendhr = 13
 1695              endif
 1696              do 600 nhr=3,iendhr
 1697                ihr = (nhr - 1) * 3
 1698
 1699                do 601 k=1,nsta
 1700
write(9,81)iday,ihr,stanam(ista(k)),
 1701             *
ws_mod(iday,nhr,ista(k)),
 1702             *
ws_obs10(iday,nhr,ista(k)),
 1703             *
ucomp_mod(iday,nhr,ista(k)),
 1704             *
ucomp_obs(iday,nhr,ista(k)),
 1705             *
vcomp_mod(iday,nhr,ista(k)),
 1706             *
vcomp_obs(iday,nhr,ista(k)),
 1707             *
press_mod(iday,nhr,ista(k)),
 1708             *
press_obs(iday,nhr,ista(k))
 1709
 1710
 1711        601     continue
 1712        600   continue
 1713              iday_old = iday
 1714              else
 1715                id00 = id00 + 1
 1716              endif
 1717        590 continue
 1718
 1719
 1720         77
format(//,17x,'Windspeed(m/s) U component
V component',
 1721             *     '   Pressure(mbars)')
 1722        78 format(1x,'Day Hour
Station Aviation Obs  Aviation Obs',
 1723             *     ' Aviation Obs
Aviation Obs')
 1724        79 format(1x,'Day Hour
Station Eta12   Obs   Eta12   ',
 1725             *     'Obs   Eta12   Obs
Eta12   Obs')
 1726        81
format(1x,i2,i4,4x,a5,3(1x,f6.2,1x,f6.2),1x,2(
1x,f8.2))
 1727
 1728
**************************************
****************
 1729
 1730        c    Call subroutine comphr.f to
calculate daily statis-
 1731        c    tics, then write to statistical
file (unit 10).
 1732        c    Write month and year header
to daily stat file.
 1733        c    690 loop is day loop.
 1734
 1735
 1736             type_ndat(1) = 'wind speed'
 1737             type_ndat(2) = 'U
component'
 1738             type_ndat(3) = 'V
```

```
component'
 1739              type_ndat(4) = 'air pressure'
 1740
 1741              iday_old = idays
 1742              do k=1,nsta
 1743                nskip(ista(k)) = 1
 1744              enddo
 1745              if(iwrit.eq.11)write(12,184)
 1746
 1747              id00 = 1
 1748
 1749              do 690 iday=idays,idayf
 1750                if(iday.ne.numd(id00))then
 1751                if(iday.gt.iday_old)then
 1752          c       write(10,*)formfd
 1753                  if(iopt_mxmn.gt.0)then
 1754                    write(81,*)formfd
 1755                  endif
 1756                endif
 1757                write(10,187)iday
 1758                if(iopt_mxmn.gt.0)then
 1759                  write(81,187)iday
 1760                endif
 1761
if(iwrit.eq.15)write(6,187)iday
 1762
 1763
if(iwrit.eq.12.and.iday.eq.iday_chck)then
 1764                  write(13,187)iday_chck
 1765                endif
 1766
 1767
!-------------------------------------------------
 1768
 1769          c     700 is the station loop for
calling comphr
 1770
 1771                icstat = 1
 1772                do 700 k=1,nsta
 1773
if(iwrit.eq.15)write(6,189)stanam(ista(k))
 1774                  if(iday.eq.idayf)then
 1775                    iendhr = 8
 1776                  else
 1777                    iendhr = 13
 1778                  endif
 1779                  n=0
 1780                  do 705 nhr=3,iendhr
 1781                    if(iday.eq.idays)then
 1782                      if(ista(k).gt.12)then
 1783                        if(nhr.lt.3)then
 1784                          write(10,188)nhr
 1785                          goto 705
 1786                        endif
 1787                      endif
 1788                    endif
 1789
 1790
if(iday.eq.idskip(nskip(ista(k)),ista(k)).and.
 1791               *
nhr.eq.icntskip(nskip(ista(k)),ista(k)))then
 1792                        if(iwrit.eq.11)then
 1793
write(12,823)stanam(ista(k)),iday,idskip
 1794               *
(nskip(ista(k)),ista(k)),nhr,icntskip
 1795               *
(nskip(ista(k)),ista(k))
 1796                        endif
 1797                        nskip(ista(k)) =
nskip(ista(k)) + 1
 1798                        n = n
 1799                      else
 1800                        n= n+1
 1801                        idat = 1
 1802                        yy(n,ista(k),idat,1) =
ws_mod(iday,nhr,
 1803               *                ista(k))
 1804                        yy(n,ista(k),idat,2) =
ws_obs10(iday,nhr,
 1805               *                ista(k))

        Program Listing 2.1 Forc.aviEta.f
(continued)
 1806                        idat = 2
 1807                      yy(n,ista(k),idat,1) =
```

```
ucomp_mod(iday,nhr,ista(k))
 1808                   yy(n,ista(k),idat,2) =
ucomp_obs(iday,nhr,ista(k))
 1809
 1810                   idat = 3
 1811                   yy(n,ista(k),idat,1) =
vcomp_mod(iday,nhr,ista(k))
 1812                   yy(n,ista(k),idat,2) =
vcomp_obs(iday,nhr,ista(k))
 1813
 1814                   idat = 4
 1815                   yy(n,ista(k),idat,1) =
press_mod(iday,nhr,ista(k))
 1816                   yy(n,ista(k),idat,2) =
press_obs(iday,nhr,ista(k))
 1817                   endif
 1818          705      continue
 1819                 ncnt = n
 1820
 1821
write(10,822)stanam(ista(k)),ncnt
 1822                 write(10,1511)
 1823                 if(iopt_mxmn.gt.0)then
 1824                   if(k.eq.istat(icstat))then
 1825
write(81,822)stanam(ista(k)),ncnt
 1826                     write(81,1511)
 1827                   endif
 1828                 endif
 1829
 1830                 ndt = 4
 1831
 1832
if(iwrit.eq.12.and.iday.eq.iday_chck)then
 1833
write(13,189)stanam(ista(k))
 1834                 endif
 1835
 1836               do 710 idat=1,ndt
 1837
if(iwrit.eq.12.and.iday.eq.iday_chck)then
 1838
write(13,*)type_ndat(idat)
 1839                 endif
 1840
 1841                 call
comphr(ista(k),idat,ncnt,rms,relerr,bias,
 1842          *
gain,corr,stderr,dif_maxpos,dif_maxneg)
 1843
 1844                 if(iwrit.eq.15)then
 1845
write(6,191)type_ndat(idat),dif_maxpos,dif_
maxneg
 1846                 endif
 1847
if(iwrit.eq.12.and.iday.eq.iday_chck)write(13
,802)
 1848
 1849                 rmswr(idat) = rms
 1850                 relerr_wr(idat) = relerr
 1851                 biaswr(idat) = bias
 1852                 gainwr(idat) = gain
 1853                 corrwr(idat) = corr
 1854                 stderwr(idat) = stderr
 1855
 1856                 difmaxpwr(idat) =
dif_maxpos
 1857                 difmaxnwr(idat) =
dif_maxneg
 1858
 1859          c      check for days with 1 or
less data points
 1860               if(ncnt.le.2)then
 1861                 rmswr(idat) = big_value
 1862                 relerr_wr(idat) =
big_value
 1863                 biaswr(idat) = big_value
 1864                 gainwr(idat) = big_value
 1865                 corrwr(idat) = big_value
 1866                 stderwr(idat) = big_value
 1867
 1868                 difmaxpwr(idat) =
big_value
 1869                 difmaxnwr(idat) =
big_value
```

```
 1870                endif
 1871
 1872           710    continue
 1873
 1874
write(10,91)(rmswr(id),id=1,ndt)
 1875
write(10,92)(relerr_wr(id),id=1,ndt)
 1876
write(10,93)(biaswr(id),id=1,ndt)
 1877
write(10,94)(gainwr(id),id=1,ndt)
 1878
write(10,95)(corrwr(id),id=1,ndt)
 1879
write(10,96)(stderwr(id),id=1,ndt)
 1880
 1881              if(iopt_mxmn.gt.0)then
 1882                if(k.eq.istat(icstat))then
 1883
write(81,91)(rmswr(id),id=1,ndt)
 1884
write(81,98)(difmaxpwr(id),id=1,ndt)
 1885
write(81,99)(difmaxnwr(id),id=1,ndt)
 1886                  icstat = icstat + 1
 1887                endif
 1888              endif
 1889
 1890           700 continue
 1891
 1892
 1893            98 format(1x,'max positive
error',5x,f8.3,3(6x,f8.3))
 1894            99 format(1x,'max negative
error',5x,f8.3,3(6x,f8.3))
 1895
 1896
!-------------------------------------------------
 1897
 1898              iday_old = iday
 1899              else
 1900                id00 = id00 + 1
 1901              endif
 1902           690 continue
 1903
 1904
 1905            96 format(1x,'standard
error',9x,f8.3,3(6x,f8.3))
 1906           184 format(/,'calculate daily
statistics')
 1907           187 format(/,1x,'For day ',i3)
 1908           188 format(1x,'First day, hr ',i3)
 1909           189 format(/,'At station ',a5)
 1910           191 format(1x,'for ',a12,', max
positive error =',
 1911                *      f6.3,/,19x,'max negative
error =',f6.3)
 1912
 1913
**************************************
****************
 1914
 1915           c     Write to cumulative file (unit
80).  Write station
 1916           c     name and number of data
points.  Write mean values
 1917           c     for windspeed, U
component, V component, and air
 1918           c     pressure, for observed and
model.
 1919
 1920
 1921              name_stat(1) = 'mean obs'
 1922              name_stat(2) = 'mean AVN'
 1923              name_stat(3) = 'standard
deviation obs'
 1924              name_stat(4) = 'standard
deviation AVN'
 1925
 1926
 1927              do 770 k=1,nsta
 1928
write(80,702)stanam(ista(k)),npt_obs(ista(k))
,
 1929                *              npt_mod(ista(k))
```

```
 1930                write(80,712)
 1931                do 775 lst=1,4
 1932
write(80,701)name_stat(lst),(rmonth_stat(ista
(k),
 1933               *                lst,n),n=1,4)
 1934           775   continue
 1935           770 continue
 1936
 1937         c     write(80,*)formfd
 1938
 1939
 1940           701 format(1x,a24,f9.2,3f13.2)
 1941           702 format(/,1x,'Station
',a5,2i9,' data points')
 1942           712
format(25x,'Windspeed(m/s) U component
V component ',
 1943               *          'pressure(mbars)')
 1944
 1945
!-------------------------------------------------
 1946
 1947         c     Call subroutine comphr.f to
calculate cumulative
 1948         c     statistics.
 1949
 1950               write(80,833)tab_title
 1951
 1952               write(80,802)
 1953               do k=1,nsta
 1954                 if(iwrit.eq.9)then
 1955
write(80,831)ista(k),stanam(ista(k)),npt_mod
(ista(k)),
 1956               *              npt_obs(ista(k))
 1957                 endif
 1958
if(npt_mod(ista(k)).ne.npt_obs(ista(k)))then
 1959
write(6,831)ista(k),stanam(ista(k)),npt_mod
 1960               *
(ista(k)),npt_obs(ista(k))
 1961                 write(6,*)'Program
stopped'
 1962                 stop
 1963               endif
 1964             enddo
 1965
 1966             if(iwrit.eq.9)write(80,829)
 1967             do 800 k=1,nsta
 1968               do n=1,npt_mod(ista(k))
 1969                 if(iwrit.eq.9)then
 1970
write(80,97)ista(k),stanam(ista(k)),n,
 1971               *
ucomp_mod_cum(ista(k),n)
 1972                 endif
 1973               enddo
 1974           800 continue
 1975
 1976
 1977             if(iwrit.eq.9)write(80,561)
 1978             do 810 k=1,nsta
 1979               nc(ista(k)) =
npt_obs(ista(k))
 1980               do n=1,npt_obs(ista(k))
 1981                 if(iwrit.eq.9)then
 1982
write(80,97)ista(k),stanam(ista(k)),n,
 1983               *
ucomp_obs_cum(ista(k),n)
 1984                 endif
 1985               enddo
 1986           810 continue
 1987
 1988
 1989
write(80,832)moc(monwr),yrc,idays,idayf
 1990
 1991
 1992             do 830 k=1,nsta
 1993
 1994               idat = 1
 1995               do n=1,npt_mod(ista(k))
 1996                 yy(n,ista(k),idat,1) =
```

```
ws_mod_cum(ista(k),n)
 1997                  yy(n,ista(k),idat,2) =
ws_obs_cum(ista(k),n)
 1998                enddo
 1999
 2000                idat = 2
 2001                do n=1,npt_mod(ista(k))
 2002                  yy(n,ista(k),idat,1) =
ucomp_mod_cum(ista(k),n)
 2003                  yy(n,ista(k),idat,2) =
ucomp_obs_cum(ista(k),n)
 2004                enddo
 2005
 2006                idat = 3
 2007                do n=1,npt_mod(ista(k))
 2008                  yy(n,ista(k),idat,1) =
vcomp_mod_cum(ista(k),n)
 2009                  yy(n,ista(k),idat,2) =
vcomp_obs_cum(ista(k),n)
 2010                enddo
 2011
 2012                idat = 4
 2013                do n=1,npt_mod(ista(k))
 2014                  yy(n,ista(k),idat,1) =
press_mod_cum(ista(k),n)
 2015                  yy(n,ista(k),idat,2) =
press_obs_cum(ista(k),n)
 2016                enddo
 2017
 2018
 2019
write(80,822)stanam(ista(k)),nc(ista(k))
 2020              write(80,1511)
 2021
 2022              do idat=1,ndt
 2023                call
comphr(ista(k),idat,nc(ista(k)),rms,relerr,
 2024             *
bias,gain,corr,stderr,difmaxpos,difmaxneg)
 2025                rmswr(idat) = rms
 2026                relerr_wr(idat) = relerr
 2027                biaswr(idat) = bias
 2028                gainwr(idat) = gain
 2029                corrwr(idat) = corr
 2030                stderwr(idat) = stderr
 2031              enddo
 2032
 2033
write(80,91)(rmswr(idat),idat=1,ndt)
 2034
write(80,92)(relerr_wr(idat),idat=1,ndt)
 2035
write(80,93)(biaswr(idat),idat=1,ndt)
 2036
write(80,94)(gainwr(idat),idat=1,ndt)
 2037
write(80,95)(corrwr(idat),idat=1,ndt)
 2038
write(80,96)(stderwr(idat),idat=1,ndt)
 2039
 2040          830 continue
 2041
 2042
 2043          829 format(/,1x,'Cumulative
Model Data')
 2044          831 format(1x,i2,2x,a5,i4,'
model pts',i4,' observed pts')
 2045          832 format(23x,a4,2x,a4,'  Days
',i2,' through',i2)
 2046          833 format(23x,a50)
 2047
 2048
**************************************
***************
 2049
 2050           91 format(1x,'rms
difference',9x,f8.3,3(6x,f8.3))
 2051           92 format(1x,'relative
error',9x,f8.3,3(6x,f8.3))
 2052           93
format(1x,'bias',19x,f8.3,2(6x,f8.3),5x,f9.3)
 2053           94
format(1x,'gain',19x,f8.3,3(6x,f8.3))
 2054           95 format(1x,'correlation
coefficient',f8.3,3(6x,f8.3))
 2055           97 format(1x,i2,2x,a5,i4,f8.2)
```

```
 2056
 2057          563 format(' Station    id  idskip
icnt  icntskip')
 2058          564
format(1x,a5,2x,2(1x,i5),1x,2(1x,i5))
 2059
 2060          802 format(/)
 2061          822 format(/,1x,'Station
',a5,8x,i4,' Data Pts')
 2062          823 format(1x,a5,2x,4i4)
 2063
 2064         1501
format(1x,a5,i10,f10.2,2x,f8.2)
 2065         1511
format(23x,'Windspeed(m/s) U component
V component ',
 2066           *       'pressure(mbars)')
 2067
 2068             stop
 2069
 2070         3455 continue
 2071             write(6,*)' nij is greater than
isize'
 2072
 2073             end
```

Subroutines :

Readlatlon - For the Eta12 option, readlatlon reads latitude and longitude data from a netcdf file. The dimensions of alat and alon are in reverse order from alat_Eta and alon_Eta, the latitude/longitude variables in the main program.

Disphr - Calculates distances, in kilometers, from grid cells to station locations. Disphr takes into account the curvature of the Earth.

Readuvp - For the Eta12 option, readuvp reads values of U component and V component of wind, and air pressure from a netcdf file. Values are read from a grid of nx (longitude) by ny (latitude). The array dimensions are reversed from the array dimensions which appear in the main program.

Uvcomp - Observed wind is read as windspeed and direction. Uvcomp converts windspeed and direction (degrees) into U and V components. The subroutine shifts the wind direction by 180 degrees in order to convert from meteorological convention to normal.

Sigma - For each station, for both model and observed data, sigma is called to calculate the standard deviation. When sigma is called, all values, model and observed, have already been stored in arrays. The number of data points, model and observed, are also stored in npt_mod and npt_obs, respectively.

Dotprod - Given the U component of the observed wind, the U component of the model wind, the V component of the observed wind, and the V component of the model wind, dotprod will calculate the speed of the model wind in the direction of the observed wind.

Comphr - This subroutine calculates the rms error, relative error, bias, gain, correlation coefficient, and the standard error. All values, model and observed, are stored in the array variable yy. Comphr also determines max and min information for each data type. Comphr calls subroutine calstat to calculate the mean, the standard deviation, and the coefficient of variation for any set of data points.

Program Listing A.2 Forc.avieta.f: Subrountines

```
      subroutine readlatlon(iwr,filename,nx,ny,alat,alon)

c lf95 readwindpressnc.f -o readwindpressnc.x   -L$OQCSBIN -loqcs -I/usr/local/include -L/usr/local/lib -lnetcdf


c  readlatlon : Reads latitude and longitude data from
c  Eta netcdf file.  The dimensions of alat and alon are
c  in reverse order from alat_Eta and alon_Eta in the main
c  program.


****************************************************************

      character*60 filename
      real*4 alat(ny,nx),alon(ny,nx)

C Netcdf reading variables
      include 'netcdf.inc'

      integer NCID,STATUS,NVARS,NGATTS,UNLIMDIMID
      integer IDVAR,COUNT(3)
      integer dimids(3),ndims,nnEtanodes,NTS
      integer base_date(4)


****************************************************************

      if(iwr.eq.1)then
        write(6,31)filename
        write(6,*)'nx =',nx,'ny = ',ny
      endif


      STATUS = NF_OPEN(filename,NF_NOWRITE,NCID)
if(STATUS.NE.NF_NOERR)write(6,*)'Problem NF_OPEN'

c  Extract dimensions and compare against fortran declarations
      STATUS = NF_INQ_VARID(NCID,'sugrd',IDVAR)
      STATUS = NF_INQ_VARNDIMS(NCID,IDVAR,ndims)

      if(ndims.ne.3)then
        write(6,32)filename,ndims

                Subroutine Readlatlon
        stop
      endif

      status = NF_INQ_VARDIMID(NCID,IDVAR,dimids)
      status = NF_INQ_DIMLEN(NCID,dimids(3),NTS)


      STATUS = NF_INQ_VARID(NCID,'lat',IDVAR)
      STATUS = NF_GET_VAR_REAL(NCID,IDVAR,alat)
      STATUS = NF_INQ_VARID(NCID,'lon',IDVAR)
      STATUS = NF_GET_VAR_REAL(NCID,IDVAR,alon)

      STATUS = NF_CLOSE(NCID)
```

```
  56
**************************************
***************
  57
  58    31 format(/,'Enter subroutine
readlatlon :',
  59       *      /,'Latitudes and longitudes read
from Eta file ',
  60       *      /,a60)
  61    32 format(/,'Netcdf file ',a60,' bad
dimensions ',i9)
  62
  63       return
  64       end
```

Subroutine Disphr

```
   1       SUBROUTINE
DISPHR(XLT1,XLON1,XLT2,XLON2,D)
   2
   3  C    This subroutine will calculate the
distance (km)
   4  C    between two points, given the
latitude and longitude
   5  C    of each.  The curvature of the
Earth is accounted for.
   6  C
   7  C    Variables :
   8  C
   9  C    Input -   XLT1     Latitude of grid
cell
  10  C              XLON1    Longitude of grid
cell
  11  C              XLT2     Latitude of station
  12  C              XLON2    Longitude of
station
  13  C
  14  C    Output -  D        Distance (km)
  15
  16       REAL LA0,LA1,LO0,LO1,LB,LL
  17
  18       HAV(X)=(SIN(.5*X))**2
  19       AHAV(X)=2.*ASIN(SQRT(X))
  20       CONV=57.29578
  21
  22       LA0=XLT1/CONV
  23       LO0=XLON1/CONV
  24       LA1=XLT2/CONV
  25       LO1=XLON2/CONV
  26       LL=LA0+LA1
  27       LB=LA0-LA1
  28
  29
R=AHAV(HAV(LB)+COS(LA0)*COS(LA1
)*HAV(LO0-LO1))
  30       D=6371.0 * R
  31
  32       RETURN
  33       END
```

Subroutine Disphr

```
   1       subroutine
readuvp(iwrt,filename,ntime,jday,nx,ny,ucom
,vcom,press)
   2
   3  c  readuvp : Reads U component, V
component, and air pressure
   4  c  data from Eta netcdf files.  The array
dimensions are
   5  c  reversed from the array dimensions
which appear in the
   6  c  main program.
   7  c  Call is made to subroutine Gregorian
to convert from
   8  c  Julian day to calendar day.
   9
  10
**************************************
***************
  11
  12  c Input arguments :
  13  c       iwrt - iwrite
  14  c    filename - Eta netcdf filename
  15  c       ntime - number of time values to
read u, v, and
  16  c               pressure values for
  17  c         nx - number of longitude values
```

```
to read from
  18 c              file (grid dimension)
  19 c          ny - number of latitude values
  20
  21
  22       integer NTIME
  23
  24       character*60 filename
  25
  26       real*4
ucom(ny,nx,ntime),vcom(ny,nx,ntime),press
  27      *          (ny,nx,ntime)
  28       real*8 jday(ntime)
  29
  30       common/chck/optmodel
  31  C Netcdf reading variables
  32       include 'netcdf.inc'
  33
  34       integer
NCID,STATUS,NVARS,NGATTS,UNLIM
DIMID
  35       integer IDVAR,COUNT(3)
  36       integer
dimids(3),ndims,nnEtanodes,NTS
  37       integer base_date(4)
  38       integer iyr
  39
  40       real*4 time(ntime)
  41       real*8 j1,julian

              Subroutine Readuvp
  42       real*8 yr,month,day,hour
  43       real*8 ryr,rmonth,rday,rhour,minute
  44
  45
**************************************
***************
  46
  47       if(iwrt.eq.14)then
  48         write(6,31)
  49         write(6,32)filename
  50         write(6,*)'ntime = ',ntime
  51         write(6,*)'nx =',nx,'ny = ',ny
  52       endif
  53
  54       STATUS =
NF_OPEN(filename,NF_NOWRITE,NCID)
  55
if(STATUS.NE.NF_NOERR)write(6,*)'Prob
lem NF_OPEN'
  56
  57  c  Extract dimensions and compare
against fortran declarations
  58       STATUS =
NF_INQ_VARID(NCID,'sugrd',IDVAR)
  59       STATUS =
NF_INQ_VARNDIMS(NCID,IDVAR,ndim
s)
  60
  61       if(ndims.ne.3)then
  62         write(*,*)'Netcdf file ',filename,'
bad dimensions ',
  63      *            ndims
  64         stop
  65       endif
  66
  67       status =
NF_INQ_VARDIMID(NCID,IDVAR,dimid
s)
  68       status =
NF_INQ_DIMLEN(NCID,dimids(1),NTS1)
  69       status =
NF_INQ_DIMLEN(NCID,dimids(2),NTS2)
  70       status =
NF_INQ_DIMLEN(NCID,dimids(3),NTS3)
  71
  72       if(NTS3.gt.NTIME)then
  73         write(*,*)'netcdf file ',filename,'
too big',NTS3,'>',NTIME
  74         stop
  75       endif
  76
  77       NTIME=NTS3
  78
  79       STATUS =
NF_INQ_VARID(NCID,'sugrd',IDVAR)
```

```
 80       STATUS =
NF_GET_VAR_REAL(NCID,IDVAR,ucom
)
 81       STATUS =
NF_INQ_VARID(NCID,'svgrd',IDVAR)
 82       STATUS =
NF_GET_VAR_REAL(NCID,IDVAR,vcom
)
 83       STATUS =
NF_INQ_VARID(NCID,'prmsl',IDVAR)
 84       STATUS =
NF_GET_VAR_REAL(NCID,IDVAR,press
)
 85
 86
 87 c   Reconstruct the julian day
 88       STATUS =
NF_INQ_VARID(NCID,'time',IDVAR)
 89       STATUS =
NF_GET_VAR_REAL(NCID,IDVAR,time)
 90       STATUS =
NF_GET_ATT_INT(NCID,IDVAR,'base_da
te',base_date)
 91
 92
if(STATUS.NE.NF_NOERR)write(*,*)'Prob
lem GET_ATT basedate'
 93       if(iwrt.eq.14)write(6,*)
'base_date',base_date
 94       yr=base_date(1)
 95
 96       month=base_date(2)
 97       day=base_date(3)
 98       hour=base_date(4)
 99
100
j1=julian(iwrt,yr,month,day,hour)
101
102              if(iwrt.eq.14)write(6,33)j1
103              call
gregorian(iwrt,j1,ryr,rmonth,rday,rhour,minu
te,second)
104              if(iwrt.eq.14)then
105
write(6,34)j1,ryr,rmonth,rday,rhour,minute
106              endif
107
108              if(iwrt.eq.14)write(6,36)
109              do n=1,NTIME
110                jday(n) = j1 + time(n)
111
if(iwrt.eq.14)write(6,35)n,jday(n),j1,time(n)
112              enddo
113
114
**************************************
***************
115
116          31 format(//,'Enter readuvp to
read u component, v component,'
117               *      /,'and air pressure from
netcdf file')
118          32 format(/,1x,'Eta file ',a72)
119          33 format(1x,'Julian date (base
date) is ',f13.5)
120          34 format(1x,f12.4,6f9.2)
121          35 format(1x,i4,2f13.3,f10.3)
122          36 format(/)
123
124
**************************************
***************
125
126        c    write(6,*)optmodel
127        c    stop
128             return
129             end
130
131
132             FUNCTION
JULIAN(iwr,yr,month,day,hour)
133             real*8 yr,month,day,hour
134             real*8 Y, m ,julian
135
136
if(iwr.eq.14)write(6,1)yr,month,day,hour
```

```
 137              Y = yr
 138              if(yr.lt.100.
.and.yr.gt.50.)then
 139                Y = yr+1900.
 140              elseif(yr.lt.100.
.and.yr.le.50.)then
 141                Y = yr+2000.
 142              endif
 143
 144              if(month.le.2.)then
 145                Y = Y-1
 146                m = month +12.
 147              else
 148                Y = Y
 149                m = month
 150              endif
 151
 152              JULIAN = aint(365.25*Y) +
aint(30.6001*(m+1))
 153             &       + day + hour/24. +
1720981.50
 154
 155
 156            1 format(1x,'yr month day
hour ',f7.1,3f6.2)
 157
 158              END
```

```
  1       subroutine
uvcomp(ws,wd,ucom,vcom,drplt)
  2
  3   c Purpose : To convert wind speed and
wind direction
  4   c         (in degrees) to a U component
and a V component.
  5   c         Also, to change wind direction
by 180 deg
  6   c         (from meteorological
convention to normal).
  7   c
  8   c Author : Phil Richardson
  9   c
 10   c Date : September 4, 1997
 11
 12
**************************************
**********
 13
 14   c    Input arguments :
 15
 16   c    ws - observed wind speed at 10
meters
 17   c    wd - wind direction
(meteorological convention,
 18   c         direction from which it is
blowing.
 19
 20
 21        parameter(pi=3.141593)
 22
 23
 24        pi2 = pi * 2.0
 25
 26        wdrad = wd/57.296
 27
 28        if(wdrad.ge.0.0.and.wdrad.lt.pi)then
 29          drplt = wdrad + pi
 30        endif
 31        if(wdrad.ge.pi.and.wdrad.lt.pi2)then
 32          drplt = wdrad - pi
 33        endif
 34        drplt = drplt * 57.296
 35
 36   c    (corrected)
 37        ucom = ws * sin(wdrad)
 38        vcom = ws * cos(wdrad)
 39
 40        return
 41        end
```

Subroutine Uvcomp

```
  1       subroutine
sigma(ns,ndt,iwr,dat_typ,stdev)
  2
  3
```

```
c Purpose : To calculate standard
deviation
c
c Author : Phil Richardson
c
c Date : March 9, 1998


****************************************
****************

c
c     Inpt arguments :
c
c          ns - station
c         ndt - ndat; windspeed, U comp,
V comp, air pressure
c         iwr - iwrit
c     dat_typ - observed or model


      parameter (nstt=12,npts=744)

      character*3 dat_typ


common/sigdat/npt_obs(nstt),ws_obs_cum(n
stt,npts),
     *
ucomp_obs_cum(nstt,npts),vcomp_obs_cum
(nstt,npts),
     *
press_obs_cum(nstt,npts),ws_mod_cum(nstt,
npts),
     *
ucomp_mod_cum(nstt,npts),vcomp_mod_cu
m(nstt,npts),
     *
press_mod_cum(nstt,npts),npt_mod(nstt)


****************************************
****************

c     Calculate mean value

      valtot = 0.0

      if(dat_typ.eq.'obs')then
      if(ndt.eq.1)then

if(iwr.eq.13)write(13,91)npt_obs(ns)
      endif
      do lp=1,npt_obs(ns)
        if(ndt.eq.1)valtot = valtot +
ws_obs_cum(ns,lp)

                Subroutine Sigma
        if(ndt.eq.2)valtot = valtot +
ucomp_obs_cum(ns,lp)
        if(ndt.eq.3)valtot = valtot +
vcomp_obs_cum(ns,lp)
        if(ndt.eq.4)valtot = valtot +
press_obs_cum(ns,lp)
      enddo
      valavg = valtot/npt_obs(ns)
      endif


      if(dat_typ.eq.'mod')then
      if(ndt.eq.1)then

if(iwr.eq.13)write(13,92)npt_mod(ns)
      endif
      do lp=1,npt_mod(ns)
        if(ndt.eq.1)valtot = valtot +
ws_mod_cum(ns,lp)
        if(ndt.eq.2)valtot = valtot +
ucomp_mod_cum(ns,lp)
        if(ndt.eq.3)valtot = valtot +
vcomp_mod_cum(ns,lp)
        if(ndt.eq.4)valtot = valtot +
press_mod_cum(ns,lp)
      enddo
      valavg = valtot/npt_mod(ns)
      endif
```

```
!-----------------------------------------------------
  c    Calculate standard deviation.  stdev is standard
  c    deviation.

       if(dat_typ.eq.'obs')then
       sqrsum = 0.0
       do 100 lp=1,npt_obs(ns)
         if(ndt.eq.1)sqrterm = (ws_obs_cum(ns,lp)-valavg)**2
         if(ndt.eq.2)sqrterm = (ucomp_obs_cum(ns,lp)-valavg)**2
         if(ndt.eq.3)sqrterm = (vcomp_obs_cum(ns,lp)-valavg)**2
         if(ndt.eq.4)sqrterm = (press_obs_cum(ns,lp)-valavg)**2
         sqrsum = sqrsum + sqrterm
   100 continue
       var = sqrsum/float(npt_obs(ns))
       endif

       if(dat_typ.eq.'mod')then
       sqrsum = 0.0
         do 110 lp=1,npt_mod(ns)

        Subroutine Sigma (continued)
         if(ndt.eq.1)sqrterm = (ws_mod_cum(ns,lp)-valavg)**2
         if(ndt.eq.2)sqrterm = (ucomp_mod_cum(ns,lp)-valavg)**2
         if(ndt.eq.3)sqrterm = (vcomp_mod_cum(ns,lp)-valavg)**2
         if(ndt.eq.4)sqrterm = (press_mod_cum(ns,lp)-valavg)**2
         sqrsum = sqrsum + sqrterm
   110   continue
         var = sqrsum/float(npt_mod(ns))
       endif


       stdev = sqrt(var)

*****************************************
****************

   91 format(1x,i4,' observed data points')
   92 format(1x,i4,' model data points')
  101 format(//)


            return
            end


      subroutine dotprod(ucobs,ucmod,vcobs,vcmod,wsdot)

c Purpose : To calculate the speed of the model wind
c         (Eta or GFS) in the direction of the observed
c         wind.
c
c Author : Phil Richardson
c
c Date : December 15, 2004


*****************************************
****************

c    Input arguments :

c    ucobs - U component, observed
c    ucmod - U component, model
c    vcobs - V component, observed
c    vcmod - V component, model


*****************************************
****************

      parameter(nstt=12,nhrs=8,ndys=31)
```

```
      common/debug/iwrt

      wspd = 0.01

      wspdmod = sqrt(ucmod**2 +
vcmod**2)
      wspdobs = sqrt(ucobs**2 +
vcobs**2)

      if(wspdobs.gt.wspd)then
        wsdot = (ucobs*ucmod +
vcobs*vcmod)/sqrt(ucobs**2 +
     *          vcobs**2)
      else
        wsdot = wspdmod - wspdobs
        if(iwrt.eq.16)then

write(6,*)ucobs,ucmod,vcobs,vcmod
          write(6,*)wsdot
        endif
      endif


      return
      end

      subroutine
comphr(ns,idt,nc,rms,re,b,g,rp,stderp,dfmaxp
,dfmaxn)

c     Model time series .....
yy(npts,ns,idt,1)
c     Observ time series .....
yy(npts,ns,idt,2)
c     sm(1),sd(1),cv(1) mean, std dev, cf
var --- series 1
c     sm(2),sd(1),cv(1) mean, std dev, cf
var --- series 2
c
c      Input arguments :
c       ns - station
c       iwr - iwrit
c
c      Output arguments :
c      rms --- rms difference
c         re --- dimensionless relative error
(0-1)
c
c         b --- bias
c        g --- gain
c      rp --- correlation coefficient
c     stderp -- modified standard error of
the estimate

c     Final version date (with corected
calculations of
c     correlation coefficient and standard
error :
c     April 8, 1998.


****************************************
***************

      parameter(npts=744,nstatot=12)

      character*5 stanm(nstatot)

      dimension dif(npts)

      common/debug/iwr

common/statpr/yy(npts,nstatot,4,2),iday,iday
_chck,
     *          stanm
      common/statp/smp(2),sdp(2),cvp(2)

      double precision
c1,c2,sumx1,sumy1,sumx2,sumxy1,gdbl,
     *     bdbl


****************************************
***************
```

```
      data small/0.00000001/

      if(nc.eq.0)return

        sum=0.
        sumx1=0.
        sumx2=0.
        sumy1=0.
        sumxy1=0.


if(iwr.eq.12.and.iday.eq.iday_chck)then
          idebug = 1
        else
          idebug = 0
        endif
        if(idebug.eq.1)then
          write(13,1701)

write(13,1702)stanm(ns),iwr,idt,nc
        endif

        dfmaxp = -10.0
        dfmaxn =  10.0
        if(idebug.eq.1)write(13,1707)
        do n=1,nc
          if(idebug.eq.1)then

write(13,1708)yy(n,ns,idt,1),yy(n,ns,idt,2)
          endif

dif(n)=yy(n,ns,idt,1)-yy(n,ns,idt,2)
          if(dif(n).gt.dfmaxp)then
            dfmaxp = dif(n)
          endif
          if(dif(n).lt.dfmaxn)then
            dfmaxn = dif(n)
          endif
          if(iwr.eq.15)then
            write(6,1713)dfmaxp
            write(6,1714)dfmaxn
          endif

          sum=sum+dif(n)*dif(n)
          sumx1=sumx1+yy(n,ns,idt,1)
          sumy1=sumy1+yy(n,ns,idt,2)

sumx2=sumx2+yy(n,ns,idt,1)*yy(n,ns,idt,1)

sumxy1=sumxy1+yy(n,ns,idt,1)*yy(n,ns,idt,2
)
        enddo
        if(idebug.eq.1)then

write(13,1709)sumx1,sumy1,sumx2,sumxy1
        endif

        call
calstat(yy(1,ns,idt,1),nc,0,sm,sd,cv)

        smp(1)=sm
        sdp(1)=sd
        cvp(1)=cv
        call
calstat(yy(1,ns,idt,2),nc,0,sm,sd,cv)

        smp(2)=sm
              sdp(2)=sd
              cvp(2)=cv

              denom=1./float(nc)
              denom2=1./float(nc-2)
              if(idebug.eq.1)then
                write(13,1703)denom
                write(13,1704)denom2
              endif


              rms=sqrt(sum*denom)
              xm = sumx1 * denom
              om=sumy1*denom
              summ=0.

              do n=1,nc
```

```
sumo=abs(om-yy(n,ns,idt,2)) +
abs(om-yy(n,ns,idt,1))
 118
summ=summ+sumo*sumo
 119              enddo
 120              re=sum/summ
 121
 122
 123         c    Normal Equations - Linear
Regression
 124
 125              c1=sumxy1 -
sumx1*sumy1*denom
 126              c2=sumx2 -
sumx1*sumx1*denom
 127              if(idebug.eq.1)then
 128                write(13,1712)c1,c2
 129              endif
 130
 131         c    Calculate gain and bias.
 132              gdbl = c1/(c2 + small)
 133              bdbl = sumy1*denom -
gdbl*sumx1*denom
 134              g = gdbl
 135              b = bdbl
 136              if(idebug.eq.1)then
 137                write(13,1711)g,b
 138              endif
 139
 140
!---------------------------------------------
 141
 142         c    modified correlation
coefficient and standard error
 143
 144              sumrt = 0.0
 145              sumse1 = 0.0
 146              sumse2 = 0.0
 147
 148              do n=1,nc
 149                sumrt = sumrt +
(yy(n,ns,idt,1)-xm) *
 150             *        (yy(n,ns,idt,2)-om)
 151              sumse1 = sumse1 +
(yy(n,ns,idt,1)-xm)**2
 152              sumse2 = sumse2 +
(yy(n,ns,idt,2)-om)**2
 153              enddo
 154              sumrb = sumse1 * sumse2
 155              rp = sumrt/sqrt(sumrb)
 156              stder2 = (sumse2-sumrt *
sumrt/sumse1)*denom2
 157              stderp = sqrt(stder2)
 158
 159
!---------------------------------------------
 160
 161         1701 format(' entering
subroutine comphr :')
 162         1702 format(1x,a5,', iwrit =
',i3,', idt =',i2,', ncount = ',i3)
 163         1703 format(' denom = ',f7.3)
 164         1704 format(' denom2 = ',f7.3)
 165         1705 format(' stan error (radical)
= ',f7.3)
 166         1706 format(' stanard error =
',f7.3)
 167         1707 format('      model
observed')
 168         1708 format(2x,f10.3,1x,f10.3)
 169         1709 format(' sumx1 = ',f9.3,/,'
sumy1 = ',f9.3,/,
 170             *      ' sumx2 = ',f11.3,/,'
sumxy1 = ',f11.3)
 171         1711 format(' gain = ',f9.3,/,'
bias = ',f9.3)
 172         1712 format(' c1 = ',f7.3,', c2 =
',f7.3)
 173         1713 format('max positive dif =
',f5.2)
 174         1714 format('max negative dif =
',f5.2)
 175
 176              return
 177              end
 178
```

```
      SUBROUTINE CALSTAT(Y,NPT,NOFF,SM,SD,CV)

C     This subroutine will, for a given set of
C     data points, calculate the mean, the standard
C     deviation, and the coefficient of variation.
C
C     Input Arguments -
C
C       Y - Flux Values
C       NPT - Number of data points
C       NOFF - Number of data points on each end to ignore
C
C     Output Arguments-
C
C       SM - Mean
C       SD - Standard deviation
C       CV - Coefficient of variation


      PARAMETER(NTOT=744)

      DIMENSION Y(NTOT)
      DATA SMALL/1.E-10/

      SUM = 0.0
      IL=NOFF + 1
      IU=NPT -  NOFF
      NPPT=IU - IL + 1
      DO 20 I=IL,IU
   20 SUM = SUM + Y(I)

      SM = SUM/NPPT
      SUM = 0.0

      DO 30 I=IL,IU
   30 SUM = SUM + (Y(I)-SM)**2

      SD = SMALL + SQRT(SUM/(NPPT-1))
      CV = SD/SM
      RETURN
      END
```

A.2. Program Plot_wndanal.pro

The listing for plot_wndanal.pro is given in Program Listing A.2. This is an IDL program used to plot a month of observed wind data, along with points from each of the daily forecasts. From each daily forecast, four points are plotted : the start, the end, the max, and the min. The symbols used to represent forecast values include pluses, triangles, squares, and asterisks.

From the control file is read ptype, idebug, stat_name, titlnam, strttime, and endtime. Ptype is for plot type, in this case postscript. Idebug controls the debug function. Stat_name is the station name, titlnam is the plot title. Strttime and endtime specify start and end times.

Plot_wndanal.pro is a conventional IDL program. The "plot" command is used to plot the observed curve, while oplot is used to plot the forecast points. The plots are annotated with a title, station name, and a legend.

Program Listing A.3 Plot_wndanal.pro

```
1  ; Program : plot_wndanal.pro
2  ;
3  ; Purpose : This program makes use of
IDL graphics
4  ; and is written in the IDL language.
The program
5  ; plots observed time series windspeed
on one
6  ; plot (per page).  The program plots
points from
7  ; the forecast model on the same plot.
8  ;   The program contains an option to
print a legend.
9  ; For postscript ('ps') plots, there is an
option for
10 ; either landscape or portrait.
11 ; The program also contains an option
for the use of
12 ; Julian dates or calendar days to define
the time axis.
13 ;
14 ;
15 ; Language : IDL
16 ;
17 ; Version date : January 23, 2005
18 ;
19 ; Location : On CBBAY,
/disks/NASUSER/philr/dynalysis/windcom
20 ;
21 ; Author : Phil Richardson
22
23
;**************************************
***************
24
25   im = 2000
26   numdays = 31
27
28
29   filemod = ''
30   filedata = ''
31   legend = ''
32   cntrl_file = ''
33   time_axis = ''
34   y_axis = ''
35   stat_name = ''
36   ptype=''
37   plottype=''
38   rmspr = ''
39   titlnam = ''
40   time_opt = ''
41
42 ; Initialize Integer Variables
43   idebug = 0
44   nticks = 0
45   iyear = 0
46   ndays = 0
47
48 ; Dimension arrays
49   lunmod=intarr(numdays)
50   t=fltarr(im)
51
52
53   legnd = strarr(2)
54   filemodl = strarr(numdays)
55
56   wlplt = fltarr(im)
57
58   xpos=fltarr(2)
59   y1=fltarr(2)
60   x1=fltarr(2,2)
61
62   time_strt = fltarr(2)
63
64   xst=fltarr(numdays)
65   xf=fltarr(numdays)
66   yst=fltarr(numdays)
67   yf=fltarr(numdays)
68
69   xhigh=fltarr(numdays)
70   xlow=fltarr(numdays)
71   yhigh=fltarr(numdays)
72   ylow=fltarr(numdays)
73
```

```
;**************************************
***************
; Open control file, read from control
file

;      ptype - x, ps, or tek
;   idebug = 1, times (Julian dates)
;          = 2, EOF result
;          = 3, plotting of Legend
;   stat_name - station name
;    strttime - start time (Julian date)
;     endtime - end time (Julian date)
;       ndays - number of daily forecast
files to read
;      nticks - number of tick marks (time
axis)
;       iyear - year of plot
;      ilegnd - option to print legend
;       legnd - character string, for legend
;     filedat - obs data filename
;     filemod - model data filename
;    time_opt - calendar day or Julian
day
;   time_axis - time axis name
;      y_axis - Y axis name

  print,'Enter name of control file '
  read,cntrl_file
; cntrl_file = 'cnt.42035_tdl.nov02'
  openr,1,cntrl_file

          readf,1,ptype
          if(ptype eq 'ps')then begin
            readf,1,plottype
          endif
          readf,1,idebug
          readf,1,stat_name
          readf,1,titlnam
          readf,1,strttime
          readf,1,endtime
          readf,1,ndays
          ndym1 = ndays - 1
          readf,1,nticks
          readf,1,ilegnd

          readf,1,ymin,ymax,ytcks
          readf,1,time_opt
          readf,1,time_axis
          readf,1,y_axis

          if(ilegnd gt 0)then
readf,1,legend
          legnd = legend
          readf,1,filedata
          filedat = filedata

          for nd=0,ndym1 do begin
            readf,1,filemod
            filemodl(nd) = filemod
            print,filemodl(nd)
          endfor

          close,1


;-------------------------------------------------

        ; set plot type : x, ps, or tek
          set_plot,ptype

        ; set the plot scaling
          aspect=1.5
          isize = 1024
          jsize = 1200


          xs=8.0
          ys=8.0*aspect

          if(ptype eq 'ps')then begin
            if(plottype eq 'portrait')then
begin
              device, xsize=xs,$
```

```
ysize=ys,/inch,xoffs=0.25,yoffs=0.
  151             endif
  152             if(plottype eq 'landscape')then
begin
  153               device, ysize=10.0,
/landscape,$
  154                     /inches, xoffs=-2.0
  155             endif
  156           endif
  157
  158
;*************************************
**************
  159
  160         ; Open observed wind data file.
  161         ;
  162         ; Read data from OBS file
  163         ;
  164         ; variables :
  165         ;   ndatpts - number of data
points
  166
  167
  168           print,time_opt,format='("time
option is ",a4)'
  169
  170           if(idebug eq 1)then
openw,4,'time.out'
  171
  172
  173           get_lun, lun
  174           openr,lun,filedat,error=err
  175
  176           if(err ne 0) then begin
  177             print, !err_string
  178             goto, ENDPROG
  179           endif
  180
  181           print,filedat,   $
  182               format='(1x,"file",a67)'
  183
  184
;*************************************
**************
  185
  186           if(ptype eq 'x')then begin
  187
window,0,xsize=isize,ysize=jsize
  188           endif
  189           wlevel_tot = 0.0
  190           ncount = 0
  191           if(idebug eq 1)then begin
  192
printf,4,filedat,format='(1x,a64)'
  193           endif
  194
  195
;*************************************
**************
  196
  197         ; Open daily forecast files.  Loop
thru days from nd=0 to
  198         ; ndym1, read from daily forecast
files.
  199
  200           for nd=0,ndym1 do begin
  201             get_lun, lunm
  202             lunmod(nd) = lunm
  203
openr,lunmod(nd),filemodl(nd),error=err
  204             if(err ne 0)then begin
  205               print, !err_string
  206               goto, ENDPROG
  207             endif
  208             print,filemodl(nd),   $
  209                 format='(1x,"file ",a86)'
  210
  211             wlmin = ymax
  212             wlmax = ymin
  213             nptm = 0
  214
  215             READMOD:
readf,lunmod(nd),timem,wlm
  216             resultm = EOF(lunmod(nd))
  217             if(idebug eq 2)then
print,timem,wlm,resultm
```

```
 218
 219              if(resultm lt 1)then begin
 220                if(nptm eq 0)then begin
 221                  timem1 = timem
 222                  xst(nd) = timem1
 223                  yst(nd) = wlm
 224
print,timem1,format='("model file begins at
",f8.3)'
 225
print,wlm,format='("windspeed = ,",f8.3)'
 226                endif
 227
 228                if(wlm gt wlmax)then begin
 229                  xhigh(nd) = timem
 230                  wlmax = wlm
 231                endif
 232                if(wlm lt wlmin)then begin
 233                  xlow(nd) = timem
 234                  wlmin = wlm
 235                endif
 236
 237                nptm = nptm + 1
 238                goto, READMOD
 239              endif
 240
 241              if(resultm gt 0)then begin
 242                xf(nd) = timem
 243                yf(nd) = wlm
 244                if(wlm gt wlmax)then begin
 245                  xhigh(nd) = timem
 246                  wlmax = wlm
 247                endif
 248                if(wlm lt wlmin)then begin
 249                  xlow(nd) = timem
 250                  wlmin = wlm
 251                endif
 252                print,timem,format='("End
of model file reached at time",f8.3)'
 253              endif
 254
 255              print,nptm,wlmin,wlmax
 256
 257              yhigh(nd) = wlmax
 258              ylow(nd) = wlmin
 259
 260
 261              free_lun, lunm
 262            endfor
 263
 264
;**********************************
**************
 265
 266          ; Read from observed data file
 267
 268              readf,lun,time
 269              print,time,    $
 270              format='(/,1x,"file (",i1,")
starts at time =",f8.3)'
 271              point_lun,lun,0
 272
 273              READDATA:
readf,lun,time,wlevel
 274              result = EOF(lun)
 275              if(idebug eq 2)then
print,result
 276              if(time lt strttime)then goto,
READDATA
 277              if(time gt endtime)then begin
 278                ncount = ncount - 1
 279                ndatpts = ncount + 1
 280                goto, ENDLOOP
 281              endif
 282              if(ncount eq 0)then begin
 283                print,lun,time,    $
 284                format='(1x,"start time (obs)
file (",i1,") = ",f8.3)'
 285                time_strt = time
 286              endif
 287
 288              wlevel_tot = wlevel_tot +
wlevel
 289
 290
 291          ;   times (Julian dates) from year
```

```
1995, relative
 292         ;   to 1993.
 293             if(time lt 1096.0) and (time gt
731.0)then begin
 294               jd_offset = 730.0
 295             endif
 296
 297         ;   times (Julian dates) not
referenced to 1993
 298             if(time lt 366.0)then begin
 299               jd_offset = 0.0
 300             endif
 301
 302         ;   times (Julian dates) from year
1998
 303             if(time gt 1826.9)then begin
 304               jd_offset = 1826.0
 305             endif
 306
 307             time = time - jd_offset
 308             if(result lt 1)then begin
 309               if(idebug eq 1)then
printf,4,ncount,time
 310               t(ncount) = time
 311               wlplt(ncount) = wlevel
 312               ncount = ncount + 1
 313               goto, READDATA
 314             endif
 315             if(result gt 0)then begin
 316               if(idebug eq 1)then
printf,4,ncount,time
 317               print,lun,      $
 318                   format='(" End of file
(",i1,") reached")'
 319               t(ncount) = time
 320               wlplt(ncount) = wlevel
 321             endif
 322           close,lun
 323           free_lun, lun
 324
 325           ndatpts = ncount + 1
 326           ENDLOOP: print,ndatpts,      $
 327               format='(i4," data points,
End of loop")'
 328           numb_pts = ndatpts
 329
 330         ; Calculate mean
 331           rmean = wlevel_tot/ndatpts
 332
 333
 334           ncount = ndatpts - 1
 335           print,ncount,format='(/,1x,i4)'
 336
 337           print,ndatpts,format='(1x,i4)'
 338
 339         !p.multi=[0,0,1]
 340
 341
;-----------------------------------------------------
 342
 343         ; make the plot
 344
 345           !P.CHARSIZE=1.0
 346
 347           if(time_opt ne 'juld')then begin
 348             dummy =
label_date(date_format='%D')
 349           endif
 350
 351
 352         @plot01
 353
plot,t[0:ncount],wlplt[0:ncount],     $
 354             title = titlnam,          $
 355             yrange=[ymin,ymax],       $
 356             xtitle=time_axis,         $
 357             ytitle=y_axis,            $
 358             xmargin=[0,0],            $
 359             ymargin=[0,0],            $
 360             xstyle=1,ystyle=1,        $
 361             linestyle=0,              $
 362             xtickformat = 'Label_date',$
 363             xticks = nticks,          $
 364             yticks = ytcks,           $
 365
position=[0.10,0.52,0.90,0.87]
```

```
366             for nd=0,ndym1 do begin
367               if(nd eq 0)then isymb = 1
368               if(nd eq 1)then isymb = 2
369               if(nd eq 2)then isymb = 5
370               if(nd eq 3)then isymb = 6
371               if(nd eq 4)then isymb = 1
372               if(nd eq 5)then isymb = 2
373               if(nd eq 6)then isymb = 5
374               if(nd eq 7)then isymb = 6
375               if(nd eq 8)then isymb = 1
376               if(nd eq 9)then isymb = 2
377               if(nd eq 10)then isymb = 5
378               if(nd eq 11)then isymb = 6
379               if(nd eq 12)then isymb = 1
380               if(nd eq 13)then isymb = 2
381               if(nd eq 14)then isymb = 5
382               if(nd eq 15)then isymb = 6
383               if(nd eq 16)then isymb = 1
384               if(nd eq 17)then isymb = 2
385               if(nd eq 18)then isymb = 5
386               if(nd eq 19)then isymb = 6
387               if(nd eq 20)then isymb = 1
388               if(nd eq 21)then isymb = 2
389               if(nd eq 22)then isymb = 5
390               if(nd eq 23)then isymb = 6
391               if(nd eq 24)then isymb = 1
392               if(nd eq 25)then isymb = 2
393               if(nd eq 26)then isymb = 5
394               if(nd eq 27)then isymb = 6
395               if(nd eq 28)then isymb = 1
396               if(nd eq 29)then isymb = 2
397
oplot,xhigh[nd:nd],yhigh[nd:nd],psym=isym
b,symsize=1.0
398
oplot,xlow[nd:nd],ylow[nd:nd],psym=isymb,
symsize=1.0
399
oplot,xst[nd:nd],yst[nd:nd],psym=isymb,sym
size=1.0
400
oplot,xf[nd:nd],yf[nd:nd],psym=isymb,symsi
ze=1.0
401             endfor
402
403
xyouts,0.50,0.55,stat_name,size=1.5,/normal,
alignment=0.5
404
405
;**********************************
**********
406
407         ; Draw Legend
408
409          if(idebug eq 3)then begin
410
print,strttime,endtime,jd_offset
411          endif
412
413
414          ; Establish x,y coordinates for
legend
415
416           x1(0,0) = 0.38
417           y1(0) = 0.825
418           x1(0,1) = 0.46
419           y1(1) = 0.825
420           x1(1,0) = 0.61
421           x1(1,1) = 0.69
422
423           if(idebug eq 3)then begin
424             print,format='(/,3x,"For
plotting of Legend -")'
425
print,x1(0,0),x1(0,1),format='(3x,"x1 points
:",2f7.1)'
426             print,y1(0),    $
427             format='(3x,"Y position (data
coordinate) is",f7.3)'
428          endif
429
430
431          xpos(0) = 0.25
432          xpos(1) = 0.50
433          ypos = 0.83
```

```
 434          if(ilegnd gt 0)then begin
 435
xyouts,xpos(0),ypos,legnd(0),size=1.4,/NOR
MAL
 436               linesty=0
 437
plots,[x1(0,0),x1(0,1)],y1,linestyle=linesty, $
 438                    /normal
 439          endif
 440
 441
;-----------------------------------------------------
 442
 443          if(ptype eq 'ps') then
device,/close
 444
 445          if(ptype eq 'ps')then begin
 446          ;   spawn,' lp -dqms2 idl.ps'
 447          endif
 448
 449
 450          ENDPROG:
 451          end
```

APPENDIX B. ANALYSIS PROCEDURE

To perform the comparison analysis, type in forc.avi.sh. This script file will compile and run the program. There are numerous control files. Their names are of the form forc_avinn.monyr.n or forc_Etann.monyr.n, where nn is either 00 or 12, mon is a three letter abbreviation for the month, and yr is a two digit number representing the year. The following two lines provide a sample directory structure.

~/dynalysis/windcom/forc.aviEta.f
~/dynalysis/windcom/wind_plot/plot_wndanal.pro

Table B.1. Script, Source File, and Control File Inventory

Script	Source File	Example Control file
forc.avi.sh	forc.aviEta.f	forc_avi00.nov02.n
	plot_wndanal.pro	cnt.42035_00z.nov02

Listings for script and control files are provided in Appendix C. The IDL plot program does not have a script file. To run the IDL program, type idl <return>, then type .r plot_wlanal.pro <return>.

APPENDIX C. SCRIPT AND CONTROL FILES

Scripts and control files for both programs are provided below.

Forc.avi.sh

```
 PATH="/usr/local/ncarg/bin:$PATH"
 export PATH
 echo $PATH
 LD_LIBRARY_PATH="/disks/NASPUB/usr/Local/Linux/ls9560/lib:$LD_LIBRARY_PATH"
 export LD_LIBRARY_PATH
 echo $LD_LIBRARY_PATH
lf95 forc.aviETA.f comphr.f disphr.f uvcomp.f sigma.f readlatlon.f read_uvp.f gregor.f uvdot.f
calcjd.f ncrght.f -o forc.bin \
 -I/usr/local/include -L/usr/local/lib -lnetcdf
 rm *.o
# forc.bin < forc_avi00.nov02.n > phil.out
# forc.bin < forc_avi004.nov02.n > phil.out
# forc.bin < forc_avi12.nov02.n > phil.out
# forc.bin < forc_avi124.nov02.n > phil.out
# forc.bin < ETA_00z.nov02pl.n > phil.out
# forc.bin < ETA_00z.nov02.n > phil.out
# forc.bin < ETA_00z4.nov02.n > phil.out
# forc.bin < ETA_12z.nov02.n > phil.out
# forc.bin < ETA_12z4.nov02.n > phil.out
# forc.bin < ETA_00z.jan03.n > phil.out
# forc.bin < ETA_00z4.jan03.n > phil.out
# forc.bin < forc_avi00.jan03.n > phil.out
# forc.bin < forc_avi004.jan03.n > phil.out
# forc.bin < forc_avi12.jan03.n > phil.out
 forc.bin < forc_avi124.jan03.n > phil.out
# forc.bin < ETA_00z.may03.n > phil.out
# forc.bin < ETA_00z4.may03.n > phil.out
# forc.bin < forc_avi00.may03.n > phil.out
# forc.bin < forc_avi004.may03.n > phil.out
# forc.bin < forc_avi12.may03.n > phil.out
# forc.bin < forc_avi124.may03.n > phil.out
# forc.bin < forc_avi00.jul03.n > phil.out
# forc.bin < forc_avi12.jul03.n > phil.out
# forc.bin < ETA_00z.jul03.n > phil.out
# forc.bin < ETA_00z4.jul03.n > phil.out

# ctrans -d ps.mono gmETA > gmETA.ps
 lp dgom.cum.0012
```

```
  rm dgom.comp.0012
  rm dgom.stat.0012
# rm dgom.cum.0012
# rm dgom.out.00
  rm phil.out
# mv dgom.stat.0012 dgom.stat
# mv dgom.cum.0012 dgom.cum
# mv *.obstran observed_trans
# mv 42001.trans.* model_tr
# mv 42002.trans.* model_tr
# mv 42003.trans.* model_tr
# mv 42019.trans.* model_tr
# mv 42020.trans.* model_tr/ETA12/42020
# mv 42035.trans.* model_tr/ETA12/42035
# mv 42036.trans.* model_tr/ETA12/42036
# mv 42039.trans.* model_tr
# mv 42040.trans.* model_tr
# mv 42041.trans.* model_tr
# mv SRST2.trans.* model_tr
# mv PTAT2.trans.* model_tr
# rm *.obstran
# rm 42001.trans.*
# rm 42002.trans.*
# rm 42003.trans.*
# rm 42019.trans.*
# rm 42020.trans.*
# rm 42036.trans.*
# rm 42039.trans.*
# rm 42040.trans.*
# rm 42041.trans.*
# rm PTAT2.trans.*
# rm SRST2.trans.*
```

forc_avi00.nov02.n

```
aviat
00          0z or 12z forecast files
 0          iwrit
 1          idotopt
 1          itransopt
11 1 30       month, start and stop days
2002
regular
0           number of days of missing files
31
```

/disks/NASUSER/philr/dynalysis/data/atmos/model/aviation/200211/2002110100.gm
32
/disks/NASUSER/philr/dynalysis/data/atmos/model/aviation/200211/2002110200.gm
33
/disks/NASUSER/philr/dynalysis/data/atmos/model/aviation/200211/2002110300.gm
34
/disks/NASUSER/philr/dynalysis/data/atmos/model/aviation/200211/2002110400.gm
35
/disks/NASUSER/philr/dynalysis/data/atmos/model/aviation/200211/2002110500.gm
36
/disks/NASUSER/philr/dynalysis/data/atmos/model/aviation/200211/2002110600.gm
37
/disks/NASUSER/philr/dynalysis/data/atmos/model/aviation/200211/2002110700.gm
38
/disks/NASUSER/philr/dynalysis/data/atmos/model/aviation/200211/2002110800.gm
39
/disks/NASUSER/philr/dynalysis/data/atmos/model/aviation/200211/2002110900.gm
40
/disks/NASUSER/philr/dynalysis/data/atmos/model/aviation/200211/2002111000.gm
41
/disks/NASUSER/philr/dynalysis/data/atmos/model/aviation/200211/2002111100.gm
42
/disks/NASUSER/philr/dynalysis/data/atmos/model/aviation/200211/2002111200.gm
43
/disks/NASUSER/philr/dynalysis/data/atmos/model/aviation/200211/2002111300.gm
44
/disks/NASUSER/philr/dynalysis/data/atmos/model/aviation/200211/2002111400.gm
45
/disks/NASUSER/philr/dynalysis/data/atmos/model/aviation/200211/2002111500.gm
46
/disks/NASUSER/philr/dynalysis/data/atmos/model/aviation/200211/2002111600.gm
47
/disks/NASUSER/philr/dynalysis/data/atmos/model/aviation/200211/2002111700.gm
48
/disks/NASUSER/philr/dynalysis/data/atmos/model/aviation/200211/2002111800.gm
49
/disks/NASUSER/philr/dynalysis/data/atmos/model/aviation/200211/2002111900.gm
50
/disks/NASUSER/philr/dynalysis/data/atmos/model/aviation/200211/2002112000.gm
51
/disks/NASUSER/philr/dynalysis/data/atmos/model/aviation/200211/2002112100.gm
52
/disks/NASUSER/philr/dynalysis/data/atmos/model/aviation/200211/2002112200.gm
53
/disks/NASUSER/philr/dynalysis/data/atmos/model/aviation/200211/2002112300.gm
54
/disks/NASUSER/philr/dynalysis/data/atmos/model/aviation/200211/2002112400.gm

55
/disks/NASUSER/philr/dynalysis/data/atmos/model/aviation/200211/2002112500.gm
56
/disks/NASUSER/philr/dynalysis/data/atmos/model/aviation/200211/2002112600.gm
57
/disks/NASUSER/philr/dynalysis/data/atmos/model/aviation/200211/2002112700.gm
58
/disks/NASUSER/philr/dynalysis/data/atmos/model/aviation/200211/2002112800.gm
59
/disks/NASUSER/philr/dynalysis/data/atmos/model/aviation/200211/2002112900.gm
60
/disks/NASUSER/philr/dynalysis/data/atmos/model/aviation/200211/2002113000.gm
dgom.out.00
dgom.comp.0012
dgom.stat.0012
Table 1 AVN Wind Field Analysis
dgom.cum.0012
dgom.statd
3
1
11
12
305.249 start time, first forecast period
306.499 stop time, first forecast period
12 number of stations
1 station numbers
42035
2
42019
3
42020
4
42002
5
42001
6
42041
7
42040
8
42039
9
42036
10
42003
11
SRST2

```
12
 PTAT2
14
/disks/NASUSER/philr/dynalysis/data/atmos/buoy/42035/42035.nov02
29.2   latitude
94.4   longitude
5.0    wind_height
 2     iw
15
/disks/NASUSER/philr/dynalysis/data/atmos/buoy/42019/42019.nov02
27.91
95.36
5.0
 2
16
/disks/NASUSER/philr/dynalysis/data/atmos/buoy/42020/42020.nov02
26.95
96.70
5.0
 2
17
/disks/NASUSER/philr/dynalysis/data/atmos/buoy/42002/42002.nov02
25.17
94.42
10.0
 2
18
/disks/NASUSER/philr/dynalysis/data/atmos/buoy/42001/42001.nov02
25.92
89.68
10.0
 2
19
/disks/NASUSER/philr/dynalysis/data/atmos/buoy/42041/42041.nov02
27.50
90.46
 5.0
 2
20
/disks/NASUSER/philr/dynalysis/data/atmos/buoy/42040/42040.nov02
29.21
88.20
 5.0
 2
21
/disks/NASUSER/philr/dynalysis/data/atmos/buoy/42039/42039.nov02
28.80
```

86.06
5.0
2
22
/disks/NASUSER/philr/dynalysis/data/atmos/buoy/42036/42036.nov02
28.51
84.51
5.0
2
23
/disks/NASUSER/philr/dynalysis/data/atmos/buoy/42003/42003.nov02
26.01
85.91
10.0
2
24
/disks/NASUSER/philr/dynalysis/data/atmos/buoy/srst2/srst2.nov02
29.7
94.1
12.5
0
25
/disks/NASUSER/philr/dynalysis/data/atmos/buoy/ptat2/ptat2.nov02
27.8
97.1
14.94
0
42035.obstran
42019.obstran
42020.obstran
42002.obstran
42001.obstran
42041.obstran
42040.obstran
42039.obstran
42036.obstran
42003.obstran
SRST2.obstran
PTAT2.obstran
42035.trans.
42019.trans.
42020.trans.
42002.trans.
42001.trans.
42041.trans.
42040.trans.
42039.trans.

42036.trans.
42003.trans.
SRST2.trans.
PTAT2.trans.

IDL< .r plot_wndanal.pro

cnt.42035_00z.nov02

```
ps
landscape
0       idebug
!1742035!X
!17AVIATION (00Z) vs OBSERVED - WIND!X
305.0    start time
335.0    end time
30       number of daily forecast files to read
29       number of ticks
1        ilegnd
 0.00 15.00 15  yrange, and number of tick marks
cald
November 2002
m/s
!17observed!X
/disks/NASUSER/philr/dynalysis/windcom/observed_trans/42035.obstran
/disks/NASUSER/philr/dynalysis/windcom/model_tr/42035/00z/42035.trans.01
/disks/NASUSER/philr/dynalysis/windcom/model_tr/42035/00z/42035.trans.02
/disks/NASUSER/philr/dynalysis/windcom/model_tr/42035/00z/42035.trans.03
/disks/NASUSER/philr/dynalysis/windcom/model_tr/42035/00z/42035.trans.04
/disks/NASUSER/philr/dynalysis/windcom/model_tr/42035/00z/42035.trans.05
/disks/NASUSER/philr/dynalysis/windcom/model_tr/42035/00z/42035.trans.06
/disks/NASUSER/philr/dynalysis/windcom/model_tr/42035/00z/42035.trans.07
/disks/NASUSER/philr/dynalysis/windcom/model_tr/42035/00z/42035.trans.08
/disks/NASUSER/philr/dynalysis/windcom/model_tr/42035/00z/42035.trans.09
/disks/NASUSER/philr/dynalysis/windcom/model_tr/42035/00z/42035.trans.10
/disks/NASUSER/philr/dynalysis/windcom/model_tr/42035/00z/42035.trans.11
/disks/NASUSER/philr/dynalysis/windcom/model_tr/42035/00z/42035.trans.12
/disks/NASUSER/philr/dynalysis/windcom/model_tr/42035/00z/42035.trans.13
/disks/NASUSER/philr/dynalysis/windcom/model_tr/42035/00z/42035.trans.14
/disks/NASUSER/philr/dynalysis/windcom/model_tr/42035/00z/42035.trans.15
/disks/NASUSER/philr/dynalysis/windcom/model_tr/42035/00z/42035.trans.16
/disks/NASUSER/philr/dynalysis/windcom/model_tr/42035/00z/42035.trans.17
/disks/NASUSER/philr/dynalysis/windcom/model_tr/42035/00z/42035.trans.18
/disks/NASUSER/philr/dynalysis/windcom/model_tr/42035/00z/42035.trans.19
/disks/NASUSER/philr/dynalysis/windcom/model_tr/42035/00z/42035.trans.20
```

/disks/NASUSER/philr/dynalysis/windcom/model_tr/42035/00z/42035.trans.21
/disks/NASUSER/philr/dynalysis/windcom/model_tr/42035/00z/42035.trans.22
/disks/NASUSER/philr/dynalysis/windcom/model_tr/42035/00z/42035.trans.23
/disks/NASUSER/philr/dynalysis/windcom/model_tr/42035/00z/42035.trans.24
/disks/NASUSER/philr/dynalysis/windcom/model_tr/42035/00z/42035.trans.25
/disks/NASUSER/philr/dynalysis/windcom/model_tr/42035/00z/42035.trans.26
/disks/NASUSER/philr/dynalysis/windcom/model_tr/42035/00z/42035.trans.27
/disks/NASUSER/philr/dynalysis/windcom/model_tr/42035/00z/42035.trans.28
/disks/NASUSER/philr/dynalysis/windcom/model_tr/42035/00z/42035.trans.29
/disks/NASUSER/philr/dynalysis/windcom/model_tr/42035/00z/42035.trans.30