UPOS-180-02

July 15, 2005

Version 1.0

By
Linda M. Burke
Bruce A. toth
Simon P. Wing
Julia Jen
Janice M. Schofield
# Table of Contents

1. **INTRODUCTION** ........................................................................................................ 2
   1.1 Overview .................................................................................................................. 2
   1.2 Summary of Architecture ...................................................................................... 2
   1.3 Statement of Purpose ............................................................................................. 3

2. **PRODUCT STRUCTURE** ......................................................................................... 4
   2.1 Executive Script Software calling hierarchy ......................................................... 4
   2.2 Kp Forecasting Software calling hierarchy ......................................................... 4

3. **OBJECT DESCRIPTIONS** ..................................................................................... 6
   3.1 Executive Script Software ...................................................................................... 6
   3.1.1 CSH Objects .................................................................................................... 6
   3.1.2 BASH Objects .................................................................................................. 6
   3.1.3 PERL Objects .................................................................................................. 6
   3.1.4 C Objects ......................................................................................................... 8
   3.1.5 rplot Objects ................................................................................................... 9
   3.1.6 IDL Objects ..................................................................................................... 9
   3.2 Prediction Software .............................................................................................. 10
   3.2.1 Java Objects ................................................................................................. 10

4. **EXTERNAL INTERFACE** ..................................................................................... 13
   4.1 Input Data .............................................................................................................. 13
   4.2 Operating System Services .................................................................................... 13
   4.3 Output ................................................................................................................... 14
   4.4 Other software programs or libraries .................................................................... 14

5. **RELATED DOCUMENTATION** ........................................................................... 15

A. **ACRONYMS AND ABBREVIATIONS** ................................................................. 16
B. **CONTACT INFORMATION/SOURCE CODE** ...................................................... 18
C. **SAMPLE OUTPUT PLOTS** ............................................................................... 19
1 Introduction

1.1 Overview

The primary purpose of the Kp Forecast system is to provide AFWA the capability to predict Kp. Kp is a 3-hour index that is commonly used to characterize the global planetary geomagnetic disturbances caused by the solar wind over a 3-hour period. (The K indices are generally used to indicate the solar wind effects on the earth's magnetic field; the subscript "p" means "planetary" and designates a global magnetic activity index.) The official Kp index is derived using data acquired at the selected 13 ground-based magnetic observatories which lie between 46 and 63 degrees north and south geomagnetic latitude: Lerwick (UK), Eskdalemuir (UK), Hartland (UK), Ottawa (Canada), Fredericksburg (USA), Meanook (Canada), Sitka (USA), Eyrewell (New Zealand), Canberra (Australia), Lovo (Sweden), Rude Skov (Denmark), Wingst (Germany), and Witteveen (The Netherlands). Based mainly on the horizontal components of the magnetic field measured by these stations, the Kp index discriminates conservatively between the true magnetic field perturbations and the quiet-day variations.

The value of Kp ranges in 28 steps from 0 to 9, indicating very quiet to very disturbed level respectively, with fractional parts expressed in thirds of a unit. The scale is quasi-logarithmic and the integer values are sub-divided into thirds commonly expressed with symbols + and -, e.g. 0, -1, 1, 1+, 2-, 2, ...9. For many applications, it is often easier to express the fractional values explicitly without using the + and - symbols. For example, Kp values of 3+ = 3.3 = 3 and 1/3, and 5- = 4.7 = 4 and 2/3, etc.

Many models for the near-Earth space environment need Kp to predict various parameters such as atmospheric density, ionospheric conductivities, auroral particle precipitation boundaries, ionospheric and magnetospheric electric field etc. Unfortunately, Kp indices are published with significant time delay. The purpose of the Forecasting Kp project is to provide a predicted Kp using inputs from the ACE satellite.

The Extended Forecasting Kp model is based on a neural network that ingests forecasted solar wind data from the HAF code (Fry et al) and outputs a kp forecast for 1 to 4 days in advance at a 6 hour cadence.

1.2 Summary of Architecture

The system architecture is based on software that is currently used for similar purposes at JHU/APL. Figure 1.1 shows a diagram of the architecture and data flow.
Once the Kp Forecasting system is started, it runs in the background. A perl script downloads HAF data from the UAF web site, propagates and averages the data, and then feeds the data into a neural network. The results from this algorithm are then displayed on a web page along with the graphs of the IMF values and solar wind data.

1.3 Statement of purpose

The purpose of this document is to describe the software architecture and to identify the components of the system.
2 Product Structure

The Forecasting Kp system is composed of two major components. The first component is implemented in perl and IDL and managed by the Executive Script. This piece of the system runs once per day by a cron job and controls data acquisition, processing, and plotting. The second piece of the Forecasting Kp system is implemented in Java to perform the prediction process. This software runs at the command of the user as a background process.

2.1 Executive Script Software calling hierarchy

1.1 run_KpHAF
   1.1.1 haf.csh
      1.1.1.1 KPHAF_end_to_end.pl
      1.1.1.1.1 get_HAF_Data.pl
      1.1.1.1.2 HAFdata_preproc_startup
         1.1.1.1.2.1 hafdata.pro
         1.1.1.1.2.1.1 monthnum.pro
         1.1.1.1.2.1.1.1 monthnames.pro
         1.1.1.1.2.2 hafdata_propagate.pro
         1.1.1.1.2.3 hafdata_average.pro
      1.1.1.1.3 create_haf_input_files.pl
      1.1.1.1.4 merge_nn_data.pl
      1.1.1.1.5 plot_week
         1.1.1.1.5.1 week_julian_to_day.pl
         1.1.1.1.5.2 arrange_imf_labels.pl
         1.1.1.1.5.3 set_current_line.pl
         1.1.1.1.5.4 set_day_boundary.pl
         1.1.1.1.5.5 week_imf rp
         1.1.1.1.5.6 week_speed rp
         1.1.1.1.5.7 week_density rp
         1.1.1.1.5.8 timestamp rp
         1.1.1.1.5.9 week_forecast_kp rp

2.2 Extended Kp Prediction Software calling hierarchy

1.1 KpPredictor
   1.1.1 NNDataSource
   1.1.2 NNInput
      1.1.2.1 URLMappings
   1.1.3 NN_KPmodel
      1.1.3.1 NNMModel
         1.1.3.1.1 RecurrentData
         1.1.3.1.2 URLMappings
      1.1.3.2 HAFDataLoader
1.1.3.2.1 URLMappings

1.1.3.3 KpData
  1.1.3.3.1 NowcastData
    1.1.3.3.1.1 URLMappings
    1.1.3.3.1.2 DataLoader
  1.1.3.3.4 Prediction
    1.1.3.4.1 JulianTime

1.1.4 NNOutput
  1.1.4.1 URLMappings

1.1.5 NNLogs
  1.1.5.1LogFile

1.1.6 ErrorLog
3 Object Descriptions

3.1 Executive Script Software

3.1.1 CSH Objects

3.1.1.1 run_KpHAF

This executive script sources haf.csh which is necessary to run the Forecasting Kp system as a cron job.

3.1.1.2 haf.csh

This executive script sets environment variables needed to run the Extended Forecasting Kp system and copies data to the user-specified output directory.

3.1.1.3 HAF_data_preproc_startup

This script controls the idl processes which is needed in order to run the Forecasting Kp system as a cron job.

3.1.2 BASH Objects

3.1.2.1 plot_week

This script creates a GIF image plot of IMF, solar wind bulk speed, solar wind proton density, and the 4 day extended Kp forecast.

3.1.3 PERL Objects

3.1.3.1 arrange_imf_labels.pl

This script sets up labels for the output plots.

CALLING SEQUENCE: arrange_imf_labels.pl (no arguments)

3.1.3.2 create_haf_input_files.pl

This script contains the produces the real-time HAF data files needed by the Extended Forecasting Kp prediction Software.

CALLING SEQUENCE: create_haf_input_files.pl (no arguments)

3.1.3.3 get_HAF_Data.pl

This script retrieves the latest HAF data file the specified web site.
CALLING SEQUENCE: get_HAF_Data.pl (no arguments)

3.1.3.4  KPHAF_end_to_end.pl

This controls the data acquisition and processing operation of the Extended Forecasting Kp system. It calls the functions necessary to process the acquired data and create the plots of the data and predictions.

CALLING SEQUENCE: KPHAF_end_to_end.pl (no arguments)

3.1.3.5  merge_nn_data.pl

This script merges the four extended kp nn data files into one file for plotting.

CALLING SEQUENCE: merge_nn_data.pl (no arguments)

3.1.3.6  set_current_line

This script reads in the current time from a file, the minimum value for the plot, and the maximum value for the plot. It then outputs two endpoints into a file which is used by rplot in order to produce a vertical dashed line on the prediction plots to represent the current time.

CALLING SEQUENCE: set_current_line (ctime file, min, max, output file)

Where:

\[
\begin{align*}
\text{ctime file} &= \text{file to hold current time value} \\
\text{min} &= \text{minimum value for y-coordinate} \\
\text{max} &= \text{maximum value for y-coordinate} \\
\text{output file} &= \text{file to hold endpoints of line}
\end{align*}
\]

3.1.3.7  set_day_boundary.pl

This script reads in the time of day boundary from the day boundary file and outputs two endpoints for the vertical current line, according to the min - max values provided for the plot. This output is read by rplot in order to produce the vertical dotted line on the prediction plot which indicates the current time.

CALLING SEQUENCE: arrange_imf_labels.pl (dayboundary_file, min, max, dline_file)

Where:

\[
\begin{align*}
\text{dayboundary file} &= \text{file to hold time of day boundary} \\
\text{min} &= \text{minimum value for the vertical dotted line}
\end{align*}
\]
max = maximum value for the vertical dotted line
dlinet file = file to hold endpoints of line

3.1.3.8 week_julian_to_day

This script converts the Julian dates associated with the 7-day data into days of the month for the purposes of labeling the 7-day data plot.

CALLING SEQUENCE: week_julian_to_day.pl (no arguments)

RETURN: None

3.1.4 C Objects

3.1.4.1 rplot

This program produces a plot image, given a data specification and dataset.

CALLING SEQUENCE: rplot specfile.rp

Where:

specfile.rp = ASCII text file that specifies the plot format and the data to be plotted

3.1.5 rplot Objects

3.1.5.1 timestamp.rp

This ASCII text file specifies the format of the timestamp image and the location of the data to plot.

3.1.5.2 week_density.rp

This ASCII text file specifies the format of the Density plot and the location of the data to plot.

3.1.5.3 week_forecast.rp

This ASCII text file specifies the format of the Extended Kp Forecast plot and the location of the data to plot.

3.1.5.4 week_imf.rp

This ASCII text file specifies the format of the IMF plot and the location of the data to plot.
3.1.5.5  week_speed.rp  

This ASCII text file specifies the format of the Solar Wind Speed plot and the location of the data to plot.

3.1.6  IDL Objects

3.1.6.1  hafdata_average.pro

This program averages the HAF data.

   CALLING SEQUENCE: hafdata_propagate(no input parameters)

3.1.6.2  haf_data.pro

This program creates the input files needed for the nn software package.

    CALLING SEQUENCE:  haf_data(no input parameters)

3.1.6.3  hafdata_propagate.pro

This program propagates the HAF data.

    CALLING SEQUENCE: hafdata_propagate(no input parameters)

3.1.6.4  monthname.pro

This program returns a string array of month names.

    CALLING SEQUENCE: monthname(num)

    Where:

    num = optional month number

3.1.6.5  monthnum.pro

This program returns a month number given name .

     CALLING SEQUENCE: month_name(name)

     Where:

    name = month name (at least 3 characters)
3.2 Prediction Software

3.2.1 Java Objects

This section contains a summary of each Java class and the major functions used in the Extended Forecasting Kp System. A more detailed description of all attributes and methods in each class is provided in the Javadoc for this software (see Appendix B for information regarding the source code/Javadoc).

3.2.1.1 KpPredictor

This is the main class for the Extended Kp Prediction software. This class initializes all of the prediction models (one day ahead, two days ahead, three days ahead and four days ahead), and begins a looping routine. The looping routine first checks for new data, runs each model, and outputs the predictions once new data arrives. Since the new input data gets updated once a day around 2300, the predictor sleeps most of the time and wakes up around 2310 to check for new data, run each model and output the prediction. The loop will only be exited at the user’s command or upon an irreconcilable error.

3.2.1.2 NNDataSource

This class called checkNewData() method checks to if there is any new updated input HAF data.

3.2.1.3 OneDayAheadNN, TwoDayAheadNN, ThreeDayAheadNN, and FourDayAheadNN

The KpPredictor runs four different models, each contained in a separate class but all working in a similar way and extending the NNModel class. The KpPredictor initializes each of the neural network classes. Each model calls the class HAFDataLoader to load the HAF input data which is provided at the specified location. This initialization sets up a NNModel with the data to be used by the neural net, the weightfile, the transformer file, the number of nodes for the neural network, and the output transformer index value. The getPrediction() method is the only public method available from the neural network classes. The method initializes the data files and sets the values for the input and output data at the current time, calls the getInitialPrediction() method of the NNModel class, and adjusts the prediction accordingly. The getPrediction() method returns an instance of the Prediction class for that model.

3.2.1.4 NNModel

Each model’s class (e.g. OneDayAheadNNs) contains an instance of the NNModel class. This class is initialized for the specific model with the data to be
used by the neural net, the weightfile, the transformer file, the number of nodes for the neural network, and the output transformer index value. After setting up these parameters and establishing the weights and transformer values, the network is initialized for the NNModel. The description of the neural network and how it operates is beyond the scope of this document.

Two public methods are available for use by an instance of NNModel. The public method setVals(double inVals[], double outVals[]) takes the current input and output data and sets up a new RecurrentData instance. The method getInitialPrediction() uses the RecurrentData instance and the neural network to provide a prediction.

3.2.1.5 RecurrentData

This class is part of the neural network and sets up DataArrayTransformers for the input and output data for the neural network. The public methods of this class are used by the getInitialPrediction() method of the NNModel class to input values into the neural network in order to get a prediction.

3.2.1.6 Prediction

This class is used to store a prediction value and its time. The public methods getValue() and getTime() allow access to the attributes of the current prediction for a model.

3.2.1.7 HAFDataLoader

This class establishes the interface to the data file containing the current HAF data. Once the data is loaded into an array by this class, the public methods of the HAFDataLoader class provide access to each of the fields available in the data. The class stores the index of all the fields available. Data can be accessed at any time.

3.2.1.8 URLMappings

This class sets up the URL values for the data files, by reading the urls_for_data.prop file. Each entry of the data file contains a string value (e.g. “INPUT_1DAY_URL”) and the location of the data directory or file the string represents. The public method getURL(String dataName) returns the location of the data as a URL instance, given the string value of the data.

3.2.1.9 JulianTime

This class contains several static methods for use with Julian dates.
3.2.1.10 NNOutput

This class is used by the KpPredictor to output the 1-Hr and 4-Hr forecast values into separate data files. The public methods used to do this are output1HrForecast (Prediction p) and output4HrForecast(Prediction p). These output files are used by the plotting tools of the Forecasting Kp system.

3.2.1.11 ErrorLog

This class provides a single static method reportError(String error) to output error messages from the prediction software into an error file.
4 External Interface

4.1 Input Data

The Forecasting Kp system takes as input HAF data as provided by UAF.

The extended Kp Forecasting system relies on HAF input data. Below is a sample input file for HAF data. The header consists of the number of hours simulated, the year of the starting time of the simulation, the month of the starting time of the simulation, the day of the starting time of the simulation, the hours of the starting time of the simulation, the Carrington Rotation number, the AU distance of the Earth, and the magnetic field parameter type. The file includes eleven columns: The hour of the simulation, the solar wind speed, the particle density, magnetic field parameters (Bx, By, Bz, and /B/), theta and phi, the latitude of the Earth and the longitude of the Earth. The proton density and bulk speed are propagated and averaged to create the file nn_oneday.dat, nn_twoday.dat, nn_threeday.dat and nn_fourday.dat for use by the neural network. Note that the fill value for this data is –9999.9.

```
192 2005    7    9    0 2032  1.0    2
  0.00  387.66  2.28   0.76  -0.83  -0.15   1.13  -7.69
 312.54  3.73  210.87
  1.00  385.98  2.27   0.74  -0.81  -0.13   1.10  -6.54
 312.28  3.74  210.91
  2.00  384.33  2.18   0.74  -0.82  -0.10   1.11  -4.95
 312.01  3.74  210.95
  3.00  382.81  2.25   0.77  -0.87  -0.06   1.16  -3.02
 311.78  3.75  210.99
```

Below is a sample of a few lines from the file nn_oneday.dat, which is used as one of the input files to the neural network. It is the result of merging the propagated and averaged HAF data. The columns consist of the hour of simulation, the solar wind speed, the particle density, Bx, By, Bz, /B/, theta the latitude of the Earth, and the longitude of the Earth.

```
4  2005    7  13  3
  3.00  600.19  0.87  -4.63  0.74  0.04  4.69  0.41
170.95  4.15  214.77
  9.00  571.36  1.24  -4.54  1.07  -0.24  4.67  -2.88
166.69  4.17  215.01
 15.00  562.72  3.54  -3.85  3.42  -0.61  5.91  -5.00
146.18  4.20  215.24
  21.00  554.36  4.79  -4.00  1.11  0.13  4.15  1.79
164.49  4.22  215.48
```

4.2 Operating System Services

The extended forecasting Kp system, implemented in Java, will run on any machine with at least Java 1.3 installed. The shell scripts and perl scripts that supervise and handle the acquisition and processing of the data, along with the
4.3 Output

The output of the Forecasting Kp system includes:

- A current 4-Day forecast
- A log of all predictions made by each model
- GIF plots of the daily and weekly values of Bx, By, and Bz
- GIF plots of the daily and weekly values of the solar wind speed
- GIF plots of the daily and weekly values of the solar wind proton density
- GIF plots of the 4-Day forecast values

The plots are all linked into web pages, included with the system, that display the 24-Hour plots and 7-Day plots. Examples of these plots are shown in Appendix C.

4.4 Other Software Programs or Libraries

The extended forecasting Kp system was implemented in Java, and so runs on any Java-enabled operating system. The software relies on neural net library math routines, which were created in-house. Finally, an in-house Java library is used for generic functionality such as simple math, plotting, and string operations. All required libraries are provided with the extended Forecasting Kp system. A GNU gcc compiler should be present on the target system to compile the C programs.
5 Related Documentation


## Appendix A

### Acronyms and Abbreviations

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACE</td>
<td>Advanced Composition Explorer</td>
</tr>
<tr>
<td>AFWA</td>
<td>Air Force Weather Agency</td>
</tr>
<tr>
<td>APL</td>
<td>Applied Physics Laboratory of Johns Hopkins University</td>
</tr>
<tr>
<td>ASCII</td>
<td>American Standard Code for Information Interchange</td>
</tr>
<tr>
<td>Bx</td>
<td>X component of the Interplanetary Magnetic Field</td>
</tr>
<tr>
<td>By</td>
<td>Y component of the Interplanetary Magnetic Field</td>
</tr>
<tr>
<td>Bz</td>
<td>Z component of the Interplanetary Magnetic Field</td>
</tr>
<tr>
<td>GIF</td>
<td>Graphic Interchange Format</td>
</tr>
<tr>
<td>HTML</td>
<td>Hypertext Markup Language</td>
</tr>
<tr>
<td>IMF</td>
<td>Interplanetary Magnetic Field</td>
</tr>
<tr>
<td>JHU</td>
<td>Johns Hopkins University</td>
</tr>
<tr>
<td>Kp</td>
<td>Geomagnetic index that indicates the global geomagnetic disturbances</td>
</tr>
<tr>
<td>MAG</td>
<td>Magnetometer (ACE Instrument)</td>
</tr>
<tr>
<td>NOAA</td>
<td>National Oceanic and Atmospheric Administration</td>
</tr>
<tr>
<td>SWEPAM</td>
<td>Solar Wind Electron, Proton, and Alpha Monitor (ACE Instrument)</td>
</tr>
<tr>
<td>UPOS</td>
<td>University Partnering for Operational Support</td>
</tr>
<tr>
<td>USAF</td>
<td>United States Air Force</td>
</tr>
</tbody>
</table>
Appendix B  Contact Information/Source Code

Refer all science/model questions to simon.wing@jhuapl.edu.

The source code is available upon request from the UPOS Transition Team. An on-line request can be made via the feedback link on the JHU/APL UPOS web page: http://sd-www.jhuapl.edu/UPOS/index.html.
Appendix C  Sample Output Plots

Plots most recently generated on Thu Jul 14 15:58:24 UT 2005