Appendix
Verification software

Matthew Pocernich
National Center for Atmospheric Research, Boulder, Colorado, USA

Nearly every method discussed in this book will require use of a computer. This appendix studies the different aspects of software programs useful for forecast verification and discusses some options available for the practitioner. Just as the utility of verification statistics is a function of the needs of the forecast user, the value of any type of verification software is dependent on the intended use. Ultimately a program should be chosen to best meet a user's needs. This appendix discusses some general aspects of software that might be considered as well as discussing some specific options. With regard to specific software, it is telling that the authors of the different chapters of this book use a wide variety of software programs and packages. While much software has been developed for different aspects of verification, there are few universally used products. What a person ultimately chooses to use for forecast verification in part seems to depend on personal background, experience with software, what software is available, the type of software most commonly used by their peers, as well as the type of forecast they are verifying. Software programs and computing capabilities are constantly evolving and improving. The information contained in this appendix was current at the time of writing, but undoubtedly at some point it will be out of date while the concepts will retain meaning. To supplement the appendix, software and data sets used in the book will be provided via the book's website: http://emps.exeter.ac.uk/fvb.

This appendix is organized as follows. The first section discusses qualities of good software programs. Examples of these properties include accuracy, documentation, ease of use and computational efficiency. The second section discusses general categories of users, which can range from the individual researcher, to a project team, to an institution. Explicitly identifying a user's needs will often help identify the appropriate type of software. The third section discusses specific software and programming options. There are many programming and statistical programming languages used in the atmospheric science community, each with strengths and weaknesses, advocates and critics.

A.1 What is good software?

Before discussing the virtues and limitations of verification software, a moment will be spent discussing the attributes of good software in more general terms. While there are no strict definitions about what is good and bad software, the following attributes should be considered when evaluating the available tools.

A.1.1 Correctness

First and foremost, a software program or function must correctly implement the desired method. The program needs to exactly translate the intent of the method. Obviously, no one intends to create faulty software, but the more complex the software program, the more opportunities there are for errors. So how do you know if the software you are using is well written? Software engineering contains many practices designed to reduce the risk of errors, most notably testing the uses of small subroutines and documentation. Smaller, simpler functions are easier to test and verify. By building complex programs out of more simple pieces, one may reduce the chance of errors. Cleanly written code allows functions to be more easily checked. Depending on the software, this level of scrutiny may not be possible.

A.1.2 Documentation

Clear, well-written documentation ensures that a function is correctly used. This reduces the chance of user-caused errors. Furthermore, good documentation and examples can make learning a new type of software easier. The format and structure of documentation should also be considered, with a focus on cross-referenced, hyperlinked and searchable documentation.

A.1.3 Open source/closed source/commercial

Open source software allows people to use, share and modify code. There are many positive and negative attributes to open source software. With open source software, it is possible to examine the uncompiled code used to create the software. This allows you to check the application and to understand exactly how values are calculated. Even with the most thorough documentation, some details are omitted and seeing how functions are written is often the best option. In the research context, this also allows you to extend or modify the implementation of a program. Open source software is typically distributed free of charge, another significant benefit. Open source licensing protects the creator from liability due to errors. While good for the creator, the user assumes responsibility for the program being correct.

When programs are distributed as closed source, or *compiled*, it may mean that a third party has assumed some responsibility for the accuracy of the software. Depending on the context, having a company or other party assume this responsibility justifies the expense. Commercial software may also have more options for support and training, such as training courses and phone support. These features may be useful and valuable assets worth the costs, and may dictate that a user chooses a commercial product over an open source product. A downside for commercial software is the cost. Aside from the actual cost, high costs can reduce the number of users and therefore reduce the size of the community.

A.1.4 Large user base

Depending on the importance and resources of the verification project or task, it may be necessary for the analysis to be checked by an outside group or by others internally. While verification statistics can be calculated using a variety of programs with a variety of languages, it is desirable that code can be checked and reviewed. This is most easily accomplished when others understand the software and the language.

Having more users increases the chances that errors will be discovered. This occurs when users apply an algorithm to different and diverse data. Archived and searchable help lists with questions and answers are a useful resource, which benefits from a larger number of users. However, taking the number of users as a criterion for choosing a software or language has limited utility in that it is typically impossible to accurately estimate the size of a community of software users.

A.2 Types of verification users

Analogous to the idea that the value of a forecast is dependent on the users, the value of verification software is dependent on the type of user. As a user, one should explicitly consider one's needs when evaluating software options. Some general groups of people who use verification software include the following.

A.2.1 Students

For individuals new to forecast verification, simplicity and transparency take precedence over computational efficiency and sophistication. Operations performed by the code should be easily viewed and interim values produced. This allows the users to understand the methods by slowly stepping through them.

A.2.2 Researchers

Graduate students and researchers in atmospheric science may be working with new types of data, new types of forecasts and a new group of users. For this reason, it is desirable that the language be flexible enough to accept new types of forecasts, and allow the development of new verification methods.

A.2.3 Operational forecasters

Operational forecasters may have an interest in evaluating a fixed set of forecasts using a fixed suite of measures over an extended period of time. This requires that some consideration be given to forecast data archiving. For this type of user, a fixed set of methods may be sufficient and it may not be necessary to examine the code in detail.

A.2.4 Institutional use

Operational forecast centres and research institutions may have different priorities with respect to forecast verification. Institutions may select or create verification software that will exist for an extended period of time, evolve with the development of new methods, and serve multiple users. Institutions may also have more resources to devote to forecast verification. These resources may include access to software engineers who can greatly expand the capabilities of verification systems by creating customized displays and implementing new methods.

A.3 Types of software and programming languages

Programming languages describe the tools used to create programs. Low-level programming languages include Fortran, C, C+, Java and others. These languages serve as the basic building blocks for many other programs, and a skilled programmer can certainly use any of these languages to implement any of the methods discussed in this book. However, the initial effort required for the inexperienced non-programmer to get started with a low-level programming language is high. For this reason, they are not typically viewed as verification tools. Higher levels of programming languages are interpreted languages – languages that allow the user to directly enter instructions to the program by using scripts or typing at a prompt. Languages such as SAS, MATLAB, Minitab, R, IDL and NCL fall under this classification. For some of these languages such as SAS, MATLAB and Minitab, graphical user interfaces (GUIs) have been created to allow the user to operate the programs with a mouse, but processes can be automated, and functions developed and modified by creating scripts and writing code. The software and some of their properties are presented in Table A.1.

Spreadsheets such as Microsoft's Excel are operated interactively with a GUI as opposed to scripts. Scientists and statisticians may dismiss them as viable verification tools and there are a lot of reasons not to use them for research or in operations. However, Excel is probably the most commonly used program for data analysis and there is no reason to think that this is any different in the atmospheric sciences. There are certainly arguments to be made for the pedagogical advantages for using spreadsheets to teach methods to a broad audience. Finally, there are web-based technologies that could allow a user to provide forecast and observation data that would utilize code and capacity hosted on another system. However, none were found when preparing this

Table A.1 Summary of software available for forecast verification

Software/program	Key user communities	Open source[a]	Comments	Verification capabilities
Spreadsheets	Ubiquitous	Varies[b]	Arguably the most commonly used type of software for statistics	No explicit verification functions
SAS	Health sciences, business, industry	No	SAS is a dominant language used in biostatistics and medical research	Verification-related functions described in health science terminology
Minitab	Statistical process control (Six Sigma), academic	No	Frequently used to teach and implement statistical process control	Limited user-contributed functions
MATLAB	Statistics, atmospheric sciences, academic	No	Commonly used in the atmospheric science research community	No publicly available verification functions. Many members of the research community have developed personal verification libraries
R	Statistical research	Yes	Basic functionality is equivalent to other statistical programming languages. Contributed packages address a very large variety of techniques	Contain several verification explicit packages including ones that perform basic verification functions, ROC curves as well as ensemble verification
IDL	Atmospheric sciences or science related	No		Some contributed verification functions available
NCL (NCAR Command Language)	Weather and climate research	Yes	Addresses many practical issues related to weather and climate research	Designed to process climate model data Few explicit verification tools
MET (Model Evaluation Tool)	Weather forecasting and research	Yes	Well supported through online support, workshops and tutorials	Designed as a forecast verification program. While MET will work with most types and expressions of forecasts, MET is designed to work most seamlessly with WRF forecasts
EUMETCAL	Weather forecasting and research	Online		Provides tutorial on forecast verification

[a] Or available without cost.

[b] While Microsoft Excel is the most commonly used spreadsheet and not free, others open source spreadsheets are available such as Open Office.

appendix. Results of verification processes are commonly posted online and often the user may define regions and time periods, but still these tools do not yet permit the user the ability to process new data.

For the purposes of discussion, software will be classified into the following groupings: (i) spreadsheets; (ii) statistical and mathematical programming languages; (iii) institutionally supported software created by weather forecasting and research institutions; and (iv) displays of verification results.

A.3.1 Spreadsheets

Prominent statistician Brian Ripley remarked in 2002, 'Let's not kid ourselves: the most widely used piece of software for statistics is Excel'. Whether or not this is true in the atmospheric sciences can be debated, but there are certainly scientists who use spreadsheets for research. Many verification methods for point forecasts and observations can be easily implemented and possibly most easily understood using a spreadsheet. These topics include theoretical aspects of verification discussed in Chapters 2 and 3, contingency table statistics discussed in Chapter 4, and the basics of verifying continuous forecasts, considered in Chapter 5 (Sections 5.1–5.4). While the use of spreadsheets for statistics might be dismissed by scientists and statisticians as not professional or appropriate, spreadsheets allow a new user the ability to immediately see all steps in basic verification procedures. Individual forecasts can be listed by row, with forecasts, observations and errors listed in separate columns. Most spreadsheet programs such as Microsoft Excel and Open Office Calculate include basic statistical functions to help in calculating statistics such as mean square error for continuous forecasts. With the observations or events correctly coded, table or tabulate functions will provide values needed to calculate most contingency table statistics. Moderately sophisticated graphics can be created using conditional formatting, which allows a plotting symbol to be formatted based on specified conditions. Graphs also have a brush-over feature that allows a limited degree of interaction with the figures. Serious limitations for using a spreadsheet include a lack of documentation, which scripts inherently provide, data size limitations, and the additional effort required to create finished verification graphics. There are no add-ons or macros developed to directly calculate verification statistics on spreadsheet programs – though for someone with some macro programming (like Visual Basic) skills, such an implementation would not be difficult.

A.3.2 Statistical programming languages

The next level up in complexity and ability from spreadsheets consist of statistical programming languages such as SPSS, SAS, Minitab, MATLAB and R (this list of programs is by no means complete). These programs differ from spreadsheets in that they can handle much larger data sets, have more advanced statistical functions, take longer to learn and are most typically used by writing scripts. Scripts allow procedures to be documented and replicated – both important practices in good verification protocols. Working with scripts is a distinctly different environment for a user familiar with a GUI environment. Functions written in these languages can be readily shared with others who use the same language. A consideration is that the choice of language or software to some degree defines the community of people who can share and collaborate in depth when analysing data. For that reason, your choice of software may in part be dictated by the culture of the company or institution where you work. Again, most of the basic verification concepts discussed in Chapters 2, 3, 4 and 5 can be readily implemented using basic statistics functions provided in all of these packages.

Minitab

Minitab has been established as a tool used to implement the 'Six Sigma Quality Control' programs at many companies. Statistical process control (SPC) has been adopted by many companies around the world to improve quality and reduce defects. While obvious parallels may exist between problems phrased in the context of quality control and the verification of weather and climate forecasts, there is little interaction between the fields. Some verification-related functions included in Minitab are the ROC plot and basic classification statistics similar to the contingency table type statistics. Most commonly, Minitab is operated using a GUI, which some people may find easier to use than scripting languages. Minitab is sometimes used in teaching introductory statistics classes. Consequently textbooks exist, which can serve as valuable learning tools.

SAS

Within the field of biostatistics and pharmacology, SAS is the dominant statistical programming language. In part this is due to the fact that SAS is one of the languages accepted by the US Food and Drug Agency for use in the drug approval process. Because of the strong similarities between forecast verification and the evaluation of treatment effectiveness or diagnosis accuracy, SAS offers many tools and resources for forecast verification. Not only are plots such as the ROC curve (Chapter 3) included as built-in functions, there are books published by SAS Press devoted to topics such as ROC curves, sampling error and built-in functions. The documentation presents some verification techniques using a different vocabulary. For example, in health science-related fields hit rates and 1 − false alarm rates are referred to as sensitivities and specificities, respectively.

SAS is a very well-documented programming language. Procedures and methods are outlined in manuals, and SAS users often have local users' groups as well as national conferences. For these reasons, SAS should be considered as a potentially useful resource for the forecast verification community. Over the years, the similarities between medical and atmospheric science research have been pointed out by many (Gabriel, 2000), but the communities remain quite separate with limited interaction.

SAS is a commercial product. Comparatively, the SAS licence can be expensive, with the price varying depending upon the capabilities included.

MATLAB

MATLAB describes itself as a technical programming language and is widely used within the atmospheric research community. A search for the term MATLAB in the American Meteorological Society (AMS) journals returns hundreds of records, confirming its popularity within the atmospheric research community. Commonly, MATLAB is used to explore and prototype new methods and procedures. Certainly among many research groups and at many institutions there are groups of users to provide advice and support. MathWorks – the parent company of MATLAB – maintains a repository of contributed MATLAB functions. However, with the exception of a ROC plot, few contributed functions exist specifically for forecast verification. Most figures presented in Chapters 3 and 7 were produced by MATLAB using code developed by the authors.

As a commercial product, a licence is required to use MATLAB. Furthermore, basic statistical functions are not included in the base MATLAB product and require the purchase of a statistical toolbox.

IDL

IDL is another mathematical programming language that is moderately popular in the atmospheric science community. Some active members of the verification community who use IDL have contributed common code. This includes code to calculate fractional skill scores and other neighbourhood methods (Section 6.5) as well as creating Taylor diagrams (Section 5.4).

Beth Ebert of the Centre for Australian Weather and Climate Research (CAWCR) has placed links to IDL code on the CAWCR website. These include the following:

Methods for verifying Probability of Precipitation (POP) (http://www.cawcr.gov.au/projects/verification/POP3/POP3.html)

Spatial Methods and Fractional Skill Scores (http://www.cawcr.gov.au/staff/eee/#Interests)

IDL is a commercial language that requires the purchase of a licence. This limits the number of potential users and restricts the ability to share code. While examples of IDL code useful for forecast verification exist in various locations, there is no common repository for code.

NCAR Command Language (NCL)

NCL is a programming language created and supported by the National Center for Atmospheric Research (NCAR) to address the unique challenges involved in managing and handling data from climate models. NCL is an open source language that runs on most popular versions of the LinuxSoftware system. Some features of climate model data include the following: large file sizes (a climate model experiment may generate terabytes of data); a global domain – which can create some challenges with projections and multiple layers vertically that extend both in the atmosphere and ocean. NCL addresses many of these issues by relying heavily on the structure of NetCDF files (http://www.unidata.ucar.edu/software/netcdf/). On the NCL website, there are many created functions and contributed functions to address data manipulation challenges such as forecast interpolation and extrapolation. Other functions create elegant graphics that permit many projections typical of those found in the IPCC (Intergovernmental Panel on Climate Change) reports.

NCL has the basic functions needed to calculate many of the verification statistics discussed in this book. For the scientist or researcher using NCL as a primary research tool, it would be logical to extend this to the verification stage of a project. However, NCL offers little in the way of explicit climate model verification functions. There is an implementation of the Taylor diagram contributed by Karl Taylor. This lack of verification functions may in part be due to the fact that the topic of climate model verification is in its infancy (Chapter 11). Certainly, as the methods and practices mature, NCL functions will be created and contributed to the NCL community.

A link to the main directory and contributed functions for NCL is: http://www.ncl.ucar.edu/.

R statistical programming language

R is an open source programming language that over the last decade has become the dominant language in the area of statistical research. Its key features include that it is a free, open source language capable of running on all major operating systems. The base package contains functions equivalent to most statistical languages. These include functions for exploratory data analysis, linear models, graphics, matrix operations and more. While the language does permit some high level operations that support the creation of simple plots, statistical models and summaries, the language also allows functions to be used at a very low level – where any detail on a graph may be modified and options set for the calculation of a model. Most of the figures presented in Chapters 8 and 10 were produced using R software.

The truly unique part of the R language is the contributed packages. In 2010 there were over 2400 contributed packages, which address nearly every type of scientific and statistical topic imaginable. Additionally, the Bioconductor project, an open source, open development effort to create tools to further genetic research and bioinformatics, uses R as a base platform. Nearly 400 additional packages are available from this project. This is relevant to research in verification because there is a large overlap between verification challenges found in the medical scientific community and the atmospheric science community. More information about the R Statistical Programming Language can be found at www.r-project.org. There are also several very active blogs that focus on many aspects of programming in R, as well as local user groups in many locations.

An excellent introduction to the applicability of R to atmospheric and verification research is the article ‘Probabilistic weather forecasting in R’ by Fraley *et al.* (2011). This article creates a probabilistic forecast from an ensemble by employing Bayesian model averaging. Of interest to this appendix are the verification methods included in the ensembleBMA package. These methods include functions to calculate mean absolute error, continuous ranked probability score, and Brier score in a spatial context. Options exist for forecasts created with exchangeable or unique forecasts. Techniques and results are illustrated using data from published papers.

Below we highlight a few packages in R relevant to verification.

verification (NCAR, 2010) provides basic verification functions including ROC plots (Chapter 3), attribute diagrams (Chapter 7), contingency table scores (Chapters 3 and 4) and more. In this package, the verify command is an overloaded function that calculates a variety of statistics, depending on the type of forecast and observation. For example, with a binary forecast and observation, contingency table statistics are calculated.

For probabilistic forecasts and binary outcomes, scores and statistics such as the Brier score are calculated. Continuous forecasts and continuous observations may be evaluated by statistics such as the mean squared error. Ensemble forecasts of full distributions and a single outcome may be verified by continuous ranked probability scores (crps). Spatial forecasts may be evaluated using a fractional skill score or spatial intensity-scale methods.

ensembleBMA (Fraley *et al.*, 2010) provides functions designed to specifically handle the verification of ensemble forecasts. This includes crps scores, rank histograms and probability integral transform values for forecasts expressed as a continuous distribution (Chapter 8).

nnclust (Lumley, 2010) provides tools for finding nearest neighbour and minimum spanning trees. This is useful for verifying multidimensional ensemble forecasts (Chapter 8).

pROC (Robin *et al.*, 2010) displays and analyses ROC curves (Chapter 3). The following is the description of the package: 'Tools for visualizing, smoothing and comparing receiver operating characteristic (ROC curves). (Partial) area under the curve (AUC) can be compared with statistical tests based on U-statistics or bootstrap. Confidence intervals can be computed for (p)AUC or ROC curves.'

ROCR (Sing *et al.*, 2009) creates and provides a variety of presentations for the information summarized in ROC plots (Chapter 3). The following is the description for ROCR: 'ROC graphs, sensitivity/specificity curves, lift charts, and precision/recall plots are popular examples of trade-off visualizations for specific pairs of performance measures. ROCR is a flexible tool for creating cutoff-parameterized 2D performance curves by freely combining two from over 25 performance measures (new performance measures can be added using a standard interface). Curves from different cross-validation or bootstrapping runs can be averaged by different methods, and standard deviations, standard errors or box plots can be used to visualize the variability across the runs. The parameterization can be visualized by printing cutoff values at the corresponding curve positions, or by coloring the curve according to cutoff. All components of a performance plot can be quickly adjusted using a flexible parameter dispatching mechanism. Despite its flexibility, ROCR is easy to use, with only three commands and reasonable default values for all optional parameters.'

A.4 Institutional supported software

Challenges exist when verifying forecasts made across a large domain, at a high frequency or operationally where real-time verification is desired. Data handling and storage become much greater issues. In portions of this book, it is assumed that one has forecast and observations in pairs. In reality, it is often the case the forecasts and observations are different types or have different spatial and/or temporal resolution. In an operational setting, multiple users may need to view the verification statistics concurrently. In these instances, software engineering plays a greater role. Typically, this type of endeavour requires the effort and resources of an institution or team. Furthermore, an organization's mission statement and funding may limit their ability to share or support publicly available verification software.

Some examples of institutional verification systems and supported tools include the following.

A.4.1 Model Evaluation Tool (MET)

MET was developed by and is supported at the National Center for Atmospheric Research (NCAR) by the Developmental Testbed Center (DTC) with funding from the US Air Force Weather Agency and NOAA. The tool has been designed to work directly with versions of the Weather Research and Forecasting (WRF) modelling system, but can be modified to be useful for other types of forecasts. Much of the analysis presented in Chapter 6 utilized MET.

MET provides a variety of verification techniques, including:

- Standard verification scores comparing gridded model data to point-based observations.
- Standard verification scores comparing gridded model data to gridded observations.
- Spatial verification methods comparing gridded model data to gridded observations using neighbourhood, object-based and intensity-scale decomposition approaches.
- Probabilistic verification methods comparing gridded model data to point-based or gridded observations.

Another powerful addition to MET is the MODE tool. MODE provides object-based methods to identify and match forecasts and observations – matching them by treating them as geometrical features. Figure 6.8 was created using MODE.

Strengths of MET include the strong institutional support provided to method development, instruction and research. This includes biannual MET tutorials, which accompany WRF tutorials taught at NCAR. Additionally, there is an annual MET workshop that focuses on new developments in MET. The project is open source and available to researchers from most countries (subject to US trade restrictions).

MET uses a number of libraries created by other organizations to read and process data. This requires that a number of libraries be installed locally on the user's machine. While many systems are supported, installation is an issue. In general, while MET may be an extremely useful tool for people familiar with operating and conducting research in numerical weather forecasts, the system has a steep learning curve for those interested in basic verification concepts.

A.4.2 Ensemble Verification System (EVS)

EVS is an experimental prototype verification program developed by the Hydrological Ensemble Prediction (HEP) group of the US National Weather Service's Office of Hydrologic Development (OHD), (Brown *et al*., 2010). Created in Java, this program operates on most operating systems. Data need to be formatted in typical text files – one for observations, one for forecasts. Columns in these files specify the valid time and lead time for each forecast. From these files, a suite of verification plots is made including plots of root mean squared error and boxplots by pooling forecasts by observed values and errors by lead time. Without respect to lead time, ROC plots, Brier skill scores, reliability diagrams, continuous ranked probability skill scores and more are generated. EVS generates output in the form of plot files, text files and XML files. The software is well documented, but online support or a helpdesk is lacking. More information and download files can be found at http://amazon.nws.noaa.gov/ohd/evs/evs.html.

A.4.3 EUMETCAL Forecast Verification Training Module

Since 2001, the European Virtual Organisation for Meteorological Training (www.eumetcal.org) has been developing and providing training modules for different topics in the meteorological sciences. Included in these topics is a module on forecast verification. This module interactively steps through the basic concepts of forecast verification. While not technically verification software, the training module has received continued institutional support and is worth inclusion in a discussion of institutional software.

A.5 Displays of verification information

Examples of online displays of verification information exist. These sites and tools differ from verification software in that they are created to evaluate a fixed set of forecasts for a specific audience and so these tools are not suitable for research. However, they do provide good examples of how to communicate verification information to a variety of audiences. This list is limited in scope. Many organizations provide online and real-time verification information for internal use and this is frequently shared with research organizations and scientists. However, this information is not typically made available to the general public. Verification from commercial weather forecasting businesses is almost uniformly considered proprietary.

A.5.1 National Weather Service Performance Management

To evaluate all warnings and forecasts produced by the US National Weather Service, the Performance Management group has created an online verification and evaluation tool. This tool analyses a wide variety of forecasts from an online archive. Forecasts may be stratified by forecast type (e.g. fire weather, aviation, public, etc.) and by forecast product. For public forecasts, a limited number of variables may be evaluated. These include minimum and maximum temperature and probability of precipitation. Warnings that can be evaluated include storm warnings, winter weather warnings, tornado warnings and many more. A suite of statistics and scores is produced, including mean squared error. Tools are also available for downloading data from the archived information. While this site serves as a good example that verification statistics may be useful for operational forecasters, one cannot add new forecasts to the system in order to utilize the verification infrastructure. Additionally, the data contained in this system are limited to the USA. Registration is also required. (https://verification.nws.noaa.gov/).

A.5.2 Forecast Evaluation Tool

The Forecast Evaluation Tool (FET) has been created to evaluate seasonal climate forecasts for temperature and precipitation produced by the National Weather Service Climate Prediction Center. These forecasts are made for lead times ranging from $^1/_2$ month to 13 months. For a chosen lead time, this tool makes summary maps of the USA showing areas of more and less skilful forecasts. Typically, forecasts are expressed by regions where warm/normal/cool or wet/normal/dry conditions exist. Information about the confidence levels within these regions is also expressed. FET is well documented and contains tutorials to help new users interpret the verification output (http://fet.hwr.arizona.edu/ForecastEvaluationTool/).

A limitation for researchers is that new forecasts cannot be loaded into the system to take advantage of the FET's methods. The domain evaluated by FET is the continental USA.