Prescient User Manual

Hyun Lee

October 14, 2017

1 Installation

For further clarification, this quick start guide is for using Prescient 2.0 on a Windows 10 computer. All of the commands in the document are done on the Anaconda prompt.
It is important that the user installs

1. Python 3.4
2. numpy, scipy, and matplotlib
3. pyomo
4. IPOPT
5. Prescient 2.0

1.1 How to get Python

A convenient and easy way to acquire Python 3 and the other required modules is through the Anaconda Python distribution. It may be downloaded from the following website:

`https://store.continuum.io/cshop/anaconda/`

Make sure to select the graphical installer for Python 3 for Windows 32- or 64-bit, depending on your operating system. Run the executable installer to install Anaconda. This should install the relevant scientific computation libraries, numpy, scipy, and matplotlib as well as a collection of other modules.

1.2 How to get Pyomo

Installing Pyomo to Python is very simple through the Anaconda prompt
Type in on the Anaconda prompt:

```
conda install -c conda-forge pyomo
```

Then also type in:

```
conda install -c conda-forge pyomo.extras
```

Through these commands, you should have pyomo installed on your computer. To check that Pyomo is actually installed on your computer, type in:

```
pyomo --version
```

You should get something like this:

```
Pyomo 5.2 (CPython 3.6.1 on Windows 10)
```

1.3 How to get IPOPT

Getting IPOPT on Unix machine is quite simple, but getting it on Windows can be quite tricky. First, download an executable from the following website:

`http://apmonitor.com/wiki/index.php/Main/DownloadIpopt`

After downloading the executable, navigate to the Downloads folder, unzip the downloaded file, and move the folder ipopt_ampl to the C:\directory. Then add this folder to the PATH environment variable. You can do this by opening the Control Panel, navigating to System and Security, then to System, and finally clicking Advanced system settings. Click the button labeled *Environment Variables...* and then find the PATH variable in the User Variables and click *Edit...*. Then click *New* and write in *C:\ipopt_ampl*. If there is no PATH variable, click *New...* and give the variable the name PATH and the value *C:\ipopt_ampl*.
To check everything was downloaded correctly, type in:

```
ipopt --version
```

on you command line. You should get something like this:

```
Ipopt 3.9.1 (MINGW32_NT-5.1 1.0.10(0.46/3/2)), ASL(20101105)
```

1.4 Additional Setup

As a Windows user, it gets frustrating when programs work on Unix machines, but the same programs do not work for a Windows machine. Therefore to guarantee that the program works for both systems, we have to edit the registry.

1. Go to your start menu
2. Type in regedit
3. Click on Default where the data is on `"C:home\Anaconda3\python.exe" "%1"`
4. In you *value data*, add %∗ to the end.
5. Make sure to change the association of the .py files to python files.

Now you should have no trouble with getting programs from a Unix user.

2 Example

I am going to assume that the user is in the quickStart directory.
This example is simple, but very important to understanding the basic inner workings of the Prescient program.
Type in the command:
python runner.py `solar_test|` `run_populator_solar_caiso.txt`
Result:
You should be getting specific information for each date:
populator.py
Reading in data
2015-01-01: Creating scenarios.
2015-01-01: Constructing Distributions
2015-01-01: Constructing Skeleton Points
2015-01-01: Truncating Scenarios
2015-01-01: Plotting Scenarios
2015-01-01: Done creating scenarios.

2.0.1 Dependencies

To get a better idea of the relationship between the input files and the output files, read the document:
`Dependencies.docx`

3 What is Prescient?

The Prescient program is the joint outcome of Sandia labs and University of California Davis. The program was created on the vision that operations planning models can be more accurate, leading to cost-effective results. Non-parametric error densities are considered of significant importance because they effectively model the real-world scenarios of today. The output from these epi-spline basis functions are used to create probabilistic scenarios.

The program relies on non-stochastic density estimates of forecast errors by using Epi-splines functions. Epi-splines functions are piecewise functions that can be applied as long as historical error data is available. This historical error data is stored in the `solar_data.csv and raw_solar.csv` file which accounts for error estimation for each hour of the day. Thus, our scenarios are created by partitioning the error distributions and connecting the pieces of quantiles together. For example, if we want to form a low-probability tailed event, then we would have to choose quantiles that correspond to this event. Therefore the spread of the scenarios can be adjusted to allow coverage that reflects the expected domain of the local wind conditions. The specification of the hours is done by the calling of `segment_solar.txt` from sources.csv which goes into `solar_data.csv` and chooses the appropriate times.

3.1 Why Precient is Important

Because of the current computational limits of todays planning models, stochastic operations planning models will be effectively used soon. As a result, it is of utmost importance to figure out a way to construct high-quality wind power scenarios as they will be used as inputs for stochastic operations planning models. To add on, these scenarios are taken from high accuracy stochastic process models. If these constructions are successfully implemented, then we can minimize cost and maintain reliability for a broad scope of scenarios.

Currently, probability wind power scenarios rely on point forecasts. These forecasts are outputs from a numerical weather prediction model which specifies a single trajectory of wind over an operational time horizon. It is important to note that the probabilistic scenarios are constructed by sampling from parametric forms of error distributions. These wind power and forecast errors rely heavily on the geographic locations and the local wind conditions.

To add on, these scenarios should capture irregular wind patterns so that operation planning can address these problems. However, it is important for these models to differentiate between completely irregular data such as cyclic ramping events that is not actually seen in real data. Thus, probabilistic scenarios that solely rely on parametric forms of error distributions seem lacking in balancing these two attributes.

3.2 Simplified Steps of the Calculation

1. We choose the specific hours we want in the day-ahead forecast, and we calculate the estimated forecast error densities for those hours in the next-day forecast.

2. Choose a set of probability values to partition the CDF (Cumulative Distributive Function) of the error density (This part is where we start partitioning). Within each partition, we can obtain the representative forecast error based on the probability-weighted error density.

3. The error values are applied to next-day forecast for the specified hour slot which contains the information: hour and probability values

4. : Intra-hour values are interpolated to form a 24-hour scenario.

4 Terminology

1. *Day Part Separators*: DPS is a set of specific hours we use to construct the forecast error densities. These values are taken from daps directory which is one directory above quickStart.

2. *Cut or Break Point Set*: The cut points can be visualized as lines that slice DPS into pieces. More specifically, imagine for each time, t, it contains a set of k cut points from $c_1, c_2, , c_k$. Therefore, they can be specified as quantiles of the cumulative forecast error density. The directory is located one directory higher than the present directory, quickStart. The Daps directory contains all the statistical tools to solve the non-parametric, stochastic problems.

3. *Skeleton Points*: For each DPS, t, there are a set of skeleton points which represent the wind power values at t. These points individually

are representative of the forecast error partition of the two adjacent cut points. Visually, we can imagine as a dot in between two lines.

4. *Data*: The raw data is the real world wind power data taken from Bonneville Power Administration (BPA). The segmentation text files are used to determine which points of information are selected from the data.

5. *Scenarios*: The scenarios are a series of wind power quantities. The quantities at the hours associated with skeleton points are computed using error densities. The values for hours between skeleton points are calculated by interpolating the difference between bounding skeleton points and the forecast. In our case, the program created 8 scenarios with the actual forecast and the predicted forecast.

6. *png files*: The png files are graphs of each of the eight scenarios. There are nine total individual graphs, and the first graph is the overlapping of all eight scenarios, including the actual and predicted forecast.