Model Formulation

The core model consists of three sub-models: (1) the wind resource, associated land and LCOE for a given project based on the wind resource, (2) technology learning include scaling of the technology, improved performance and reduced cost, and (3) the wind project development pipeline (with co-flows for turbines, capacity and generation) and the industry value chain (turbine suppliers and project developers). We describe each of these model formulations and then in the next section, we demonstrate how we select the cases for analysis including the disaggregation into regional markets that feed the global dynamics of technology development. The model is built in Vensim® DSS for Windows Version 6.4c (x32) from Ventana Systems, Inc. Vensim is a system dynamics modeling software tool that allows a user to visually build up a system dynamics simulation by adding variables to a graphical template and then interlinking them and defining functional relationships.

Wind Resource, Cost of Energy and Policy

One of the great sources of heterogeneity and uncertainty for wind energy is the local wind resource. In this model, we aggregate the resource in a certain region into a resource "supply curve" for each state of interest. The model assumes that industry builds projects in the sites with the highest wind resources first and then incrementally less attractive sites and so on. The equation for the marginal wind resource in a given state is:

$$\begin{aligned} &Marginal\ install\ wind\ speed[States] \\ &\quad = max\ wind\ speed[States] + linear\ term\ for\ wind\ speed\ by\ land\ use[States] \\ &\quad * percent\ land\ under\ development[States] \\ &\quad + quadratic\ term\ for\ wind\ speed\ by\ land\ use[States] \\ &\quad * percent\ land\ under\ development[States]^2 \end{aligned}$$

Where *States* is the index for the current state for analysis (which may be a country or a province). The wind speed for the next best-undeveloped site is determined based on a quadratic supply curve fit to the states' wind resource data. The percent land under development is land under development or already developed over total available land in a region:[1]

$$percent\ land\ under\ development[States] = ZIDZ\left(\frac{total\ land\ under\ development[States]}{total\ potential\ land[States]}\right)$$

Other wind deployment studies, such as the US DOE Wind Vision, have used the notion of a supply curve (U.S. Department of Energy, 2015).

[1] ZIDZ is a function that stands for zero if divide by zero. It is an "if-then" statement which takes the value of the function in parentheses unless the denominator of the function is zero. Rather than use a backslash, the numerator is the first value in the expression followed by a comma and then the denominator.

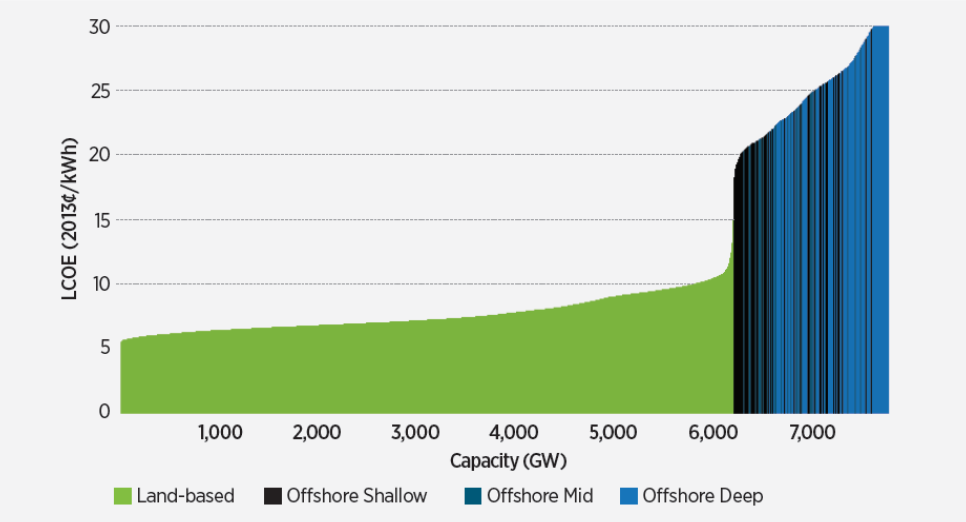

FIGURE 1: LCOE SUPPLY CURVE FOR THE ENTIRE UNITED STATES FROM THE WIND VISION STUDY THAT TAKES INTO ACCOUNT THE TECHNOLOGY COSTS, FINANCING AND ENERGY PRODUCTION FOR DIFFERENT SITES ACROSS THE UNITED STATES (U.S. DEPARTMENT OF ENERGY, 2015). NOTE THAT IT IS ESSENTIALLY A SUPERPOSITION OF TWO SUPPLY CURVES: ONE FOR OFFSHORE AND ONE FOR LAND-BASED WIND ENERGY. THIS STUDY, AS PREVIOUSLY MENTIONED, CURRENTLY ONLY LOOKS AT LAND-BASED SYSTEMS.

In that case, the study translated the supply curves to an LCOE supply curve for each region of interest based on one or more wind turbine technology configurations, the wind resource distribution in the region for all areas that are not excluded. The supply curve excludes many already developed areas, if they are sensitive regions for habitats, migration and other environmental concerns, if they are sensitive due to national security, and if they are too near population dense areas (U.S. Department of Energy, 2015).

LCOE is a project specific value that takes into account all of the upfront capital expenditures and financing costs, financing rates, the lifetime operational expenditures, and the lifetime energy production. To simplify analysis, for instance when assessing a technical innovation, LCOE is a single equation (Fingersh, Hand and Laxon, 2006):

$$LCOE = \frac{F * CAPEX + OPEX * (1 - T)}{AEP}$$

Where *F* is a fixed charge rate that reduces from the complexity of a detailed financial model, *CAPEX* is the total capital expenditures of the project including insurance, warranties and financing costs, *OPEX* is the annual operational expenditures for the project, which are reduced by a tax reduction equal to *T* the tax rate and *AEP* is the net annual energy production after all losses are taken into account.

For this model, transforming the wind resource, technology costs and financing into LCOE and it is complicated to create LCOE supply curves due to fact that many of the inputs to the LCOE equation change as the dynamics unfold. First, the technology and associated costs change as learning takes

place and this causes the expected energy production for a given resource group to vary as well. Secondly, the policies that a state undertakes to promote adoption of wind technology will change over time and affect the LCOE supply curve. So, instead of an LCOE supply curve, the model takes the technology costs, financing and wind resource and turns them into a breakeven energy production level which is compared to the marginal available expected energy production for the next highest wind resource site in a state that is still undeveloped. Next, we describe this transformation.

First, the expected annual energy for the marginal site is calculated. The model takes the wind resource data for the different states from a global GIS database of wind speed resource estimates at 50 m above ground elevation from the National Aeronautics and Space Administration (NASA) (OpenEI 2016). Since the data is at one hub height, which changes over time, a scaling of the wind speed to the current hub height is necessary. "Wind shear" is the phenomenon that controls this wind speed scaling with height above ground. Various factors can affect wind shear at a given location including where the site is in the world and characteristics of the local atmosphere as well as topography, the surrounding environment, etc. Due to lack of site-specific information at a global level, the *wind shear exponent* is assumed constant across all sites and is 1/7 or 0.143 (a commonly used value for the shear exponent over flat terrain) (AWS Scientific, 1997). The model then scales wind speed as:

$$\begin{aligned} &hub\ height\ wind\ speed[States] \\ &\quad = marginal\ install\ wind\ speed[States] \\ &\quad * hub\ height/reference\ hub\ height^{wind\ shear\ exponent} \end{aligned}$$

This turns the *hub height wind speed* into a distribution of wind speeds that represent an average year for the location. For real wind project analysis, developers often fit site data to either a Weibull or a Rayleigh distribution. In this case, we use the simpler Rayleigh distribution. The input to the distribution is the mean hub height wind speed of the site as calculated above. For each wind speed, the model calculates a probability of occurrence based on the Rayleigh distribution. The probability density function of the Rayleigh distribution is:

$$P(x;\sigma) = \frac{x}{\sigma^2} e^{-\frac{x^2}{2\sigma^2}}$$

Where *x* is the input variable, and $\sigma\sqrt{\frac{\pi}{2}}$ is the mean of the distribution and *P* is the probability of the occurrence of x. The mean in this case is then the calculated hub height wind speed for the site multiplied by $\sqrt{\frac{\pi}{2}}$ and the probability density function for wind speed becomes:

$$\begin{aligned} &Rayleigh\ wind\ speed\ probability[States, windspeed] \\ &\quad = ZIDZ\left(\frac{\pi * wind\ speed[windspeed]}{2 * hub\ height\ wind\ speed[States]^2}\right) e^{-ZIDZ\left(\frac{\pi * wind\ speed[windspeed]^2}{4 * hub\ height\ wind\ speed[States]^2}\right)} \end{aligned}$$

Where *windspeed* is the index of wind speeds from 3 m/s to 25 m/s. Most wind turbines will cut-in around 3 m/s when they overcome the force needed to rotate the blades and produce energy. For

safety reasons, most turbines will cut out at 25 m/s. The model then convolves the Rayleigh distribution with the power curve for the turbine to create an estimate of the annual energy at each wind speed, and the model then sums these to get the total annual energy before loss reductions:

$$Cumulative\ Energy\ by\ Wind\ Speed[States, windspeed] = Power\ at\ wind\ speed[States, windspeed] * Rayleigh\ wind\ speed\ probability[States, windspeed] * hours\ per\ year$$

And:

$$Potential\ annual\ energy[States] = SUM(Cumulative\ Energy\ by\ Wind\ Speed[States, windspeed!])$$

The wind turbine power curve (the power at each wind speed that the turbine is expected to produce) is determined by the machine size and design. The basic equation of power produced by a wind turbine at a given wind speed is:

$$Power\ at\ wind\ speed[States, windspeed] = 0.5 * air\ density * Rotor\ Swept\ Area * (wind\ speed[windspeed]^3) * \frac{power\ coefficient}{Watts\ per\ kW}$$

And:

$$Rotor\ Swept\ Area = \pi(\frac{Simulated\ marginal\ rotor\ diameter}{2})^2$$

This power curve equation is based on simple flow physics where power is proportional to the cubic of the wind speed, the rotor area, the air density at the site (assumed to be 1.225 kg / m^3 at sea level at 15 degrees Celsius (AWS Scientific, 1997)) and the power coefficient (which is based on machine design and has a theoretical limit of 0.59 (Manwell, 2002). Real values are less than the theoretical maximum due to lower than ideal efficiencies for the aerodynamic, mechanical and electrical sub-systems. The model uses a power coefficient that multiplies a reasonable rotor power coefficient of 0.47 by a reasonable drivetrain efficiency of 0.92 for a total of 0.43 (Fingersh, Hand and Laxon, 2006). Most turbines today limit their power output above a "rated level" so the function above is limited to the rated power of the machine and is constant for increasing wind speed once that level has been reached. The Watts per kW factor of 1000 is applied to convert the standard equation for Power from Watts to kW since that the rest of the calculations use units of kW.

The expected energy is then the potential annual energy, after losses and turbine availability are accounted for, produced by the convolution of the Rayleigh wind speed distribution above and the wind turbine power curve:

$$Expected\ annual\ energy[States] = Potential\ annual\ energy[States] * (1 - losses) * turbine\ availability$$

Once the model calculates the expected annual energy, it compares it to the breakeven annual energy through use of the wind resource distribution and estimates the percent of profitable land out of the total land in the region:

$$\begin{aligned} &percent\ profitable\ land\ out\ of\ total[States] \\ &= ZIDZ\left(\frac{expected\ annual\ energy\ max[States] - breakeven\ expected\ annual\ energy[States]}{linear\ term\ for\ wind\ speed\ by\ land\ use[States]}\right) \\ &+ ZIDZ\left(\frac{expected\ annual\ energy\ max[States] - breakeven\ expected\ annual\ energy[States]^2}{quadratic\ term\ for\ wind\ speed\ by\ land\ use[States]}\right) \end{aligned}$$

From this total, the percent of profitable land still undeveloped, the percent of profitable land under development, and the percent of land already built are determined (for use in project development portion of the model):

$$percent\ land\ under\ development\ [States] = ZIDZ\left(\frac{total\ land\ under\ development[States]}{total\ potential\ land[States]}\right)$$

And:

$$\begin{aligned} &percent\ land\ already\ built[States] \\ &= ZIDZ\left(\frac{Expected\ land\ Use\ from\ Construction\ Projects[States] + Simulated\ Land\ Use\ Installed\ Base[States]}{total\ potential\ land[States]}\right) \end{aligned}$$

And:

$$\begin{aligned} &percent\ profitable\ land\ undeveloped \\ &= percent\ profitable\ land\ out\ of\ total[States] \\ &- percent\ land\ under\ development[States] \end{aligned}$$

The diagram below shows the visual relationships of the above determination of expected annual energy and percentage of profitable sites in region.

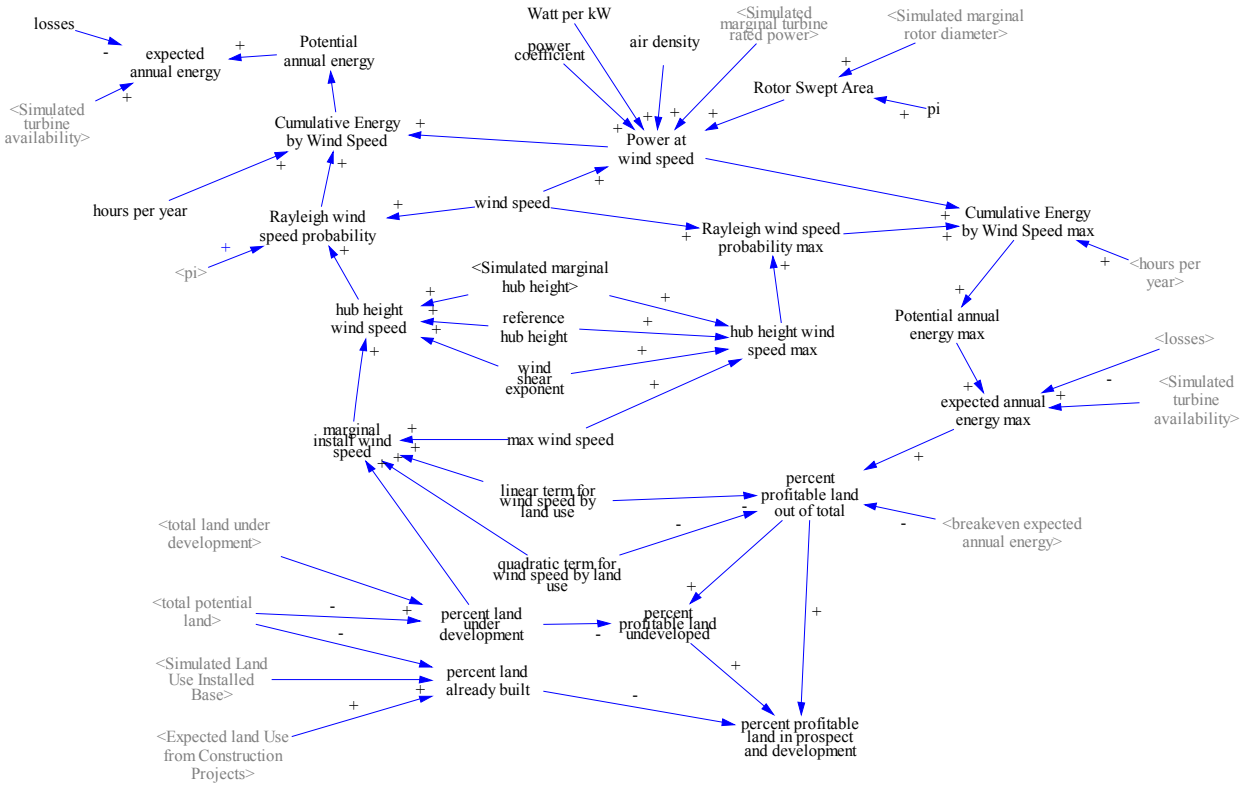

FIGURE 2: MODEL VIEW OF THE FUNCTIONS TO DETERMINE EXPECTED ANNUAL ENERGY AND THE PERCENTAGE OF PROFITABLE SITES IN A REGION BASED ON THE MARGINAL WIND RESOURCE AVAILABLE AT A SITE.

To determine breakeven annual energy, the next portion of the model considers technology, its associated costs, financing, revenues and incentives. The sub-model diagram is below and the key equation relating required energy production to revenues and costs is:

$$breakeven\ annual\ energy[States] = ZIDZ(\frac{annual\ breakeven\ costs[States]}{unit\ revenue[States]})$$

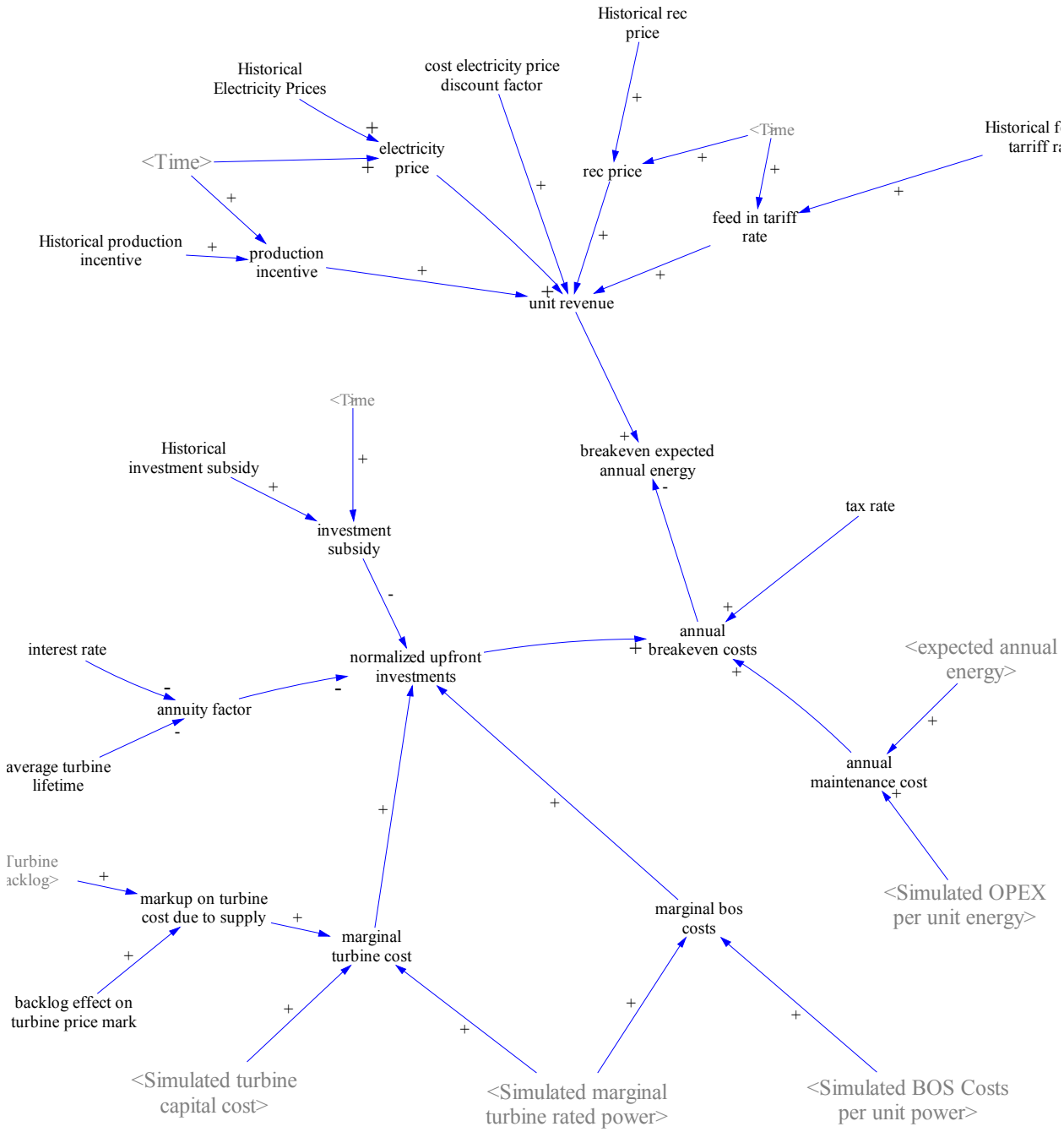

FIGURE 3: MODEL VIEW OF THE BREAKEVEN EXPECTED ANNUAL ENERGY CALCULATION.

And the constituent equations for breakeven expected annual energy include:

$$annual\ breakeven\ costs[States] = annual\ maintenance\ cost[States] * (1 - tax\ rate) + normalized\ upfront\ investments[States]$$

$$annual\ maintenance\ cost[States] = Simulated\ OPEX\ per\ unit\ energy * expected\ annual\ energy[States]$$

$$normalized\ upfront\ investments[States] = \frac{upfront\ investment\ costs[States] * (1 - investment\ subsidy[States])}{annuity\ factor}$$

The basic equations for cost of energy (including the annuity factor and reduced operations and maintenance costs for tax deductions (1 – tax rate)) for a wind plant based on NREL's wind turbine cost and scaling model (Fingersh, Hand and Laxon, 2006). The model bases the actual costs on the current technology, as we will see in the subsequent sub-section. As shown above, the normalized investment costs account for investment subsidies that are available from the state or national governments. The investment subsidies are determined from a lookup table imported from data for the region(s) of interest. Unit revenue also accounts for any available subsidies:

$$unit\ revenue[States] = max(electricity\ price[States] * cost\ electricity\ price\ discount\ factor[States] + production\ incentive[States] + rec\ price[States], feed\ in\ tariff\ rate[States])$$

Above, either there is the opportunity for the unit revenue to correspond to either a subsidized or unsubsidized price per kWh produced or the opportunity to receive a fixed feed in tariff rate if present. There can be various types of incentives on the production side including direct production incentives, renewable energy certificates (rec) prices that affect the relative value of renewable energy generation to non-renewable sources, and feed in tariffs that provide a direct price for buying electricity that is usually higher than the unsubsidized price would be. The incentives are use lookup tables imported from data for the region(s) of interest.

Lastly, there is a function for the effect of a backlog in turbine inventory on the marginal cost of the turbines that uses a lookup table. We describe the model for the wind industry supply chain in a later sub-section, but here we look at the mark-up of turbine cost due to short supply. There is evidence that the backlog in the wind industry in the late 2000's rose to 10,000 turbines or more (Kanellos, 2008) and a study found that a portion of the price mark-up during the time could be explained by higher profit margins demanded by the turbine manufacturers in the "seller's market" in addition to a growth in labor costs, warranty and other factors due to having to ramp up production quickly to meet growing demand (Bolinger and Wiser, 2011). The amount attributed to endogenous factors was roughly a 50% mark-up (Bolinger and Wiser, 2011). Thus, a look-up function applies this mark-up at the 10,000-turbine level similar to that experienced during that period.

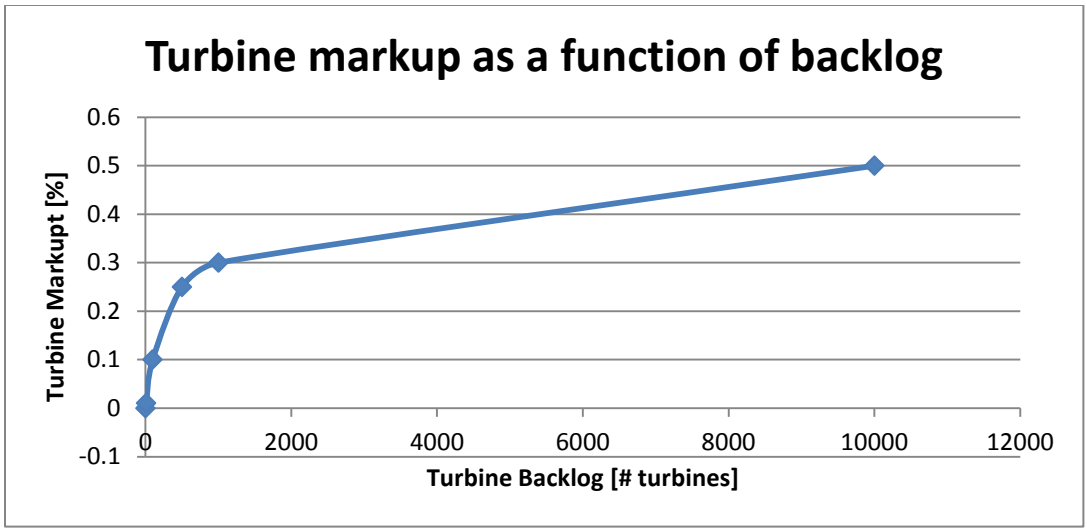

FIGURE 4: LOG FUNCTION OF TURBINE PRICE MARKUP AS A FUNCTION OF BACKLOG

Technology Innovation and Cost Trends

The major reinforcing feedback loop is the technology innovation and learning loop. There are a many number of areas of innovation in the LCOE equation for wind energy and the associated inverse breakeven annual energy production equations above. Firstly, there are performance side innovations that increase relative energy production for a given wind site. This means improved the $Power\ at\ wind\ speed$ over the various wind speeds. The factors of machine design going into the equation include the power coefficient and the rotor swept area. Thus, one obvious way of generating more electricity with a given wind turbine is to increase the rotor size. Indeed, there has been a continuous increase in rotor diameter (and area) since the early 1980's and beyond. Enabling this increase in rotor diameter over time is the technological learning from deploying more and more wind turbines. Thus, the model imposes a learning curve on rotor diameter in the model based on the total turbine installations relative to a reference value:

$$Simulated\ marginal\ rotor\ diameter = initial\ rotor\ size * ratio\ of\ total\ to\ initial\ turbine\ installation\ experience^{exponent\ rotor\ diameter}$$

Where the model uses a standard learning rate equation to calculate the learning exponent:

$$exponent\ rotor\ diameter = \frac{\ln(1\ +\ rotor\ diameter\ learning\ rate)}{\ln 2}$$

Correspondingly, the power rating of the machine has increased over time. Until very recently with a shift to lower specific power machines (power per unit area), the machine rating and rotor diameter had strong functional correlation:

$$marginal\ turbine\ power\ rating$$
$$= initial\ turbine\ power\ rating$$
$$initial\ turbine\ power\ rating * ZIDZ(\frac{Simulated\ marginal\ rotor\ diameter}{initial\ rotor\ size})^{rotor\ power\ exponent}$$
$$* ZIDZ(\frac{Simulated\ marginal\ rotor\ diameter}{initial\ rotor\ size})^{rotor\ power\ exponent}$$

However, what really matters for power produced at wind speeds below rated is the power coefficient or efficiency. The overall machine configuration (rotor diameter and rated power) affect this along with a number of aspects of the aerodynamic, mechanical and electrical design of the machine. Each aspect contributes its own efficiency at a given operating point that rolls up into the power coefficient. The power coefficient is a complex representation of the machine design and thus it is a constant at a reasonable value of 0.47 that is high for older turbines such that the model overestimates the energy production of those machines by a small amount.

Finally, the turbine availability has improved over time such that turbines break less often and the time that they are down due to each break is less. The formulation is the same as for the rotor diameter and is based on turbine installation (and thus operational) experience:

$$turbine\ availability = \frac{initial\ turbine\ availability}{ratio\ of\ total\ to\ initial\ turbine\ installation\ experience^{exponent\ availability}}$$

Where a standard learning rate equation is used to calculate the learning exponent:

$$exponent\ availability = \frac{\ln(1 + availability\ learning\ rate)}{\ln 2}$$

In addition to the learning on the performance side, there is learning associated with all aspects of the cost side of the turbine and plant. The formulation is the same for each cost element (on a per kW basis) including turbine costs, balance of station costs and operational expenditures and mimics the performance formulations based on the turbine installation and operational experience:

$$Simulated\ turbine\ capital\ cost\ per\ unit\ power$$
$$= initial\ upfront\ turbine\ costs\ per\ unit\ power$$
$$* ratio\ of\ total\ to\ initial\ turbine\ installation\ experience^{exponent\ turbine\ costs}$$

Where:

$$exponent\ turbine\ costs = \frac{\ln(1 - turbine\ costs\ learning\ rate)}{\ln 2}$$

And:

$$balance\ of\ station\ costs\ per\ unit\ power$$
$$= initial\ balance\ of\ station\ costs\ per\ unit\ power$$
$$* (ratio\ of\ total\ to\ initial\ turbine\ installation\ experience\ )\text{^}exponent\ bos\ costs$$

Where:

$$exponent\ turbine\ costs = \frac{\ln(1 - turbine\ costs\ learning\ rate)}{\ln 2}$$

And:

$$OandM\ and\ LRC\ costs\ per\ unit\ power$$
$$= initial\ annual\ OPEX\ costs\ per\ unit\ energy$$
$$* ratio\ of\ total\ to\ initial\ turbine\ installation\ experience^{exponent\ OPEX}$$

Where:

$$exponent\ OPEX = \frac{\ln(1 - turbine\ OPEX\ learning\ rate)}{\ln 2}$$

The model takes ratio of total to initial turbine installation experience from the total cumulative global turbine installation experience over a reference number of turbines installed by the year 1980. In the model analysis, we only simulate a subset of nations and states. However, learning is a global process and excluding the rest of the world turbine installation experience would lead to an underestimation of the technology scaling increases and cost reductions. Thus, the turbine supply for the rest of the world (excluding the modeled subset) is an exogenous input to the model. To do this, the python script accesses the data sheet that has cumulative turbine installations by year.

FIGURE 5: MODEL VIEW OF TECHNOLOGY AND COST LEARNING WHICH IS DRIVEN BY EXPERIENCE IN GLOBAL WIND TURBINE INSTALLATIONS.

From this, the model removes the turbine production from the modeled nations and states and then feeds the remaining turbine total to the *parameter rest of world construction rate* via a lookup table:

$$Total\ turbine\ construction\ completions = SUM(Simulated\ turbine\ construction\ finish\ rate[States!]) + rest\ of\ world\ turbine\ construction\ start\ rate$$

Where:

$$rest\ of\ world\ construction\ start\ rate = Historical\ rest\ of\ world\ turbine\ construction\ start\ rate(Time)$$

This variable for rest of world turbine construction also feeds the turbine supply chain model as well in the next sub-section. The model diagram for the learning curves for technology and cost is above.

Wind Projects and the Industry Supply Chain

The final sub-model includes the project development along with the industry dynamics. The core flow of the model is the development of projects.

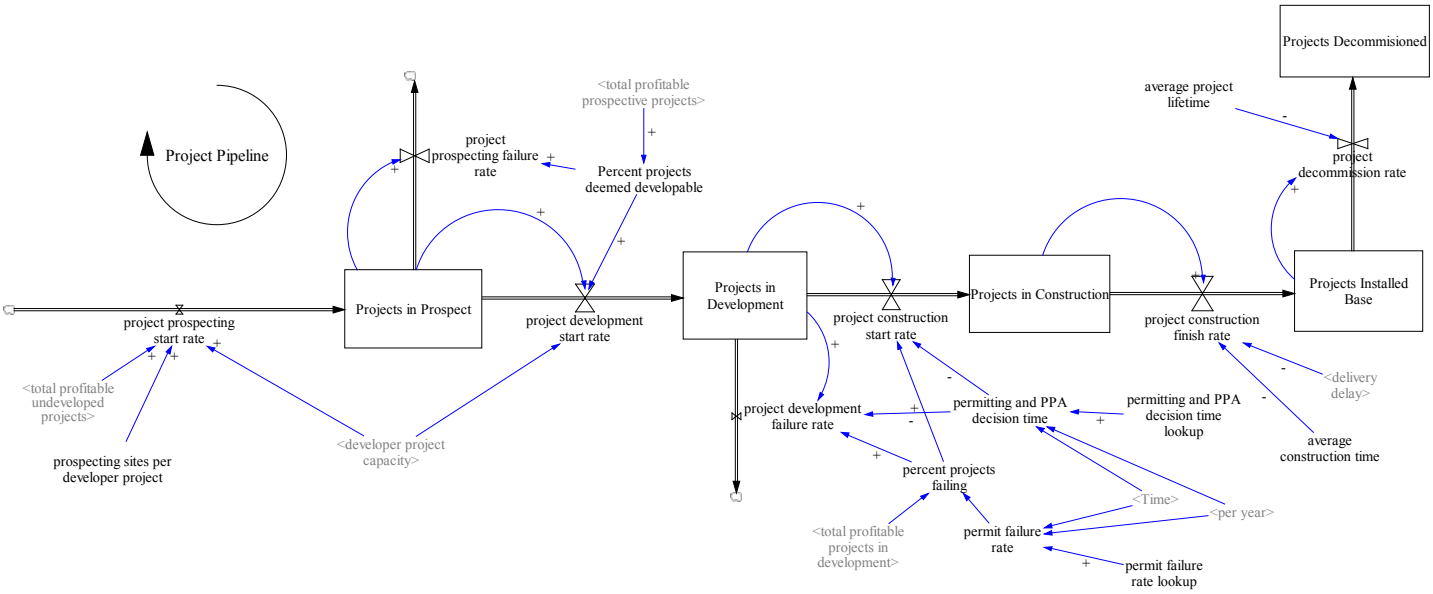

FIGURE 6: DIAGRAM OF THE MODEL PROJECT DEVELOPMENT PIPELINE.

Wind plant developers are the main drivers of the industry and develop a certain number of projects at a time (based on their size and a number of other factors). The pipeline for project development is above.

Based on the attractiveness of a given market and the relative developer capacity in that market, the developer will begin project development. Wind plant development follows a standard process including prospecting, development, construction and commissioning (the point at which the plant enters into the installed base of the wind energy sector). Developers plan on typical timelines and success rates for these projects that include (Splettstosser, 2016):

- For every one project that is built, the developer will prospect five sites.
- Of those five prospected sites, roughly four will make it to the development stage.
- Another three of those four will fail at the development stage so that only one project gets through to construction.

Thus, the model contains five *prospecting sites per developer project*, a *percent project deemed developable* of 0.8, and a *percent projects failing* in development of 0.75. Furthermore, the whole process takes about five years including one year in prospecting and four years in development for site design, leasing land, obtaining environmental and other regulatory permits, and signing agreements for interconnection, power purchase and financing (Splettstosser, 2016). Then the developer will construct the project in roughly a year or less (Splestosser interview 2016). In the early years of wind development, the timeframe for development was much shorter and there is an associated time

dependence of *Permitting and PPA decision time* on the year. The following equations are the key flows of the model's development pipeline.

Firstly, project prospecting depends on the total profitable sites available for development and the current capacity for development of regional developers:

$$\begin{aligned} &project\ prospecting\ start\ rate[States] \\ &= MIN(developer\ project\ capacity[States] \\ &* prospecting\ sites\ per\ developer\ project, total\ profitable\ undeveloped\ projects[States]) \end{aligned}$$

The next sub-section describes the developer capacity sub-model. The percent of profitable land undeveloped, the fraction of land available for wind development and the land-use per project determine the total profitable undeveloped projects:

$$\begin{aligned} &total\ profitable\ undeveloped\ projects \\ &= total\ potential\ land[States] \\ &* fraction\ of\ land\ available\ for\ wind\ development[States] \\ &* percent\ profitable\ land\ undeveloped[States]/land\ use\ per\ project \end{aligned}$$

The fraction of land available for wind development is set to a constant value of 15% in the model. Future versions of the model will include factors that make this a dynamic relationship with increasing wind energy deployment.

From the projects that go into development, a percentage will be developable and will move on to the next stages. If market conditions change and become unfavorable, then all projects will be undevelopable and released.

$$\begin{aligned} &project\ development\ start\ rate[States] \\ &= MIN(developer\ project\ capacity[States], Projects\ in\ Prospect[States] \\ &* Percent\ projects\ deemed\ developable[States]) \end{aligned}$$

$$\begin{aligned} &project\ prospecting\ failure\ rate[States] \\ &= (1 - Percent\ projects\ deemed\ developable[States]) \\ &* Projects\ in\ Prospect[States] \end{aligned}$$

For the projects in development, a certain percentage again will fail (about 75%) and that again can increase to all current projects in development if market conditions become unfavorable. Once the developer obtains all necessary permits and signs a PPA with a utility or other buyer of the electricity, typically 4 years, the successful projects will move on to construction.

$$\begin{aligned} &project\ construction\ start\ rate[States] \\ &= \frac{(1 - percent\ projects\ failing[States]) * Projects\ in\ Development[States]}{permitting\ and\ PPA\ decision\ time} \end{aligned}$$

$$project\ development\ failure\ rate[States] = \frac{Projects\ in\ Development[States] * percent\ projects\ failing[States]}{permitting\ and\ PPA\ decision\ time}$$

Projects in construction will complete on average in under a year unless there is a delivery delay due to a backlog in wind turbine inventory. The delivery delay is the total backlog over the current turbine industry production capacity.

$$project\ construction\ finish\ rate[States] = ZIDZ(\frac{Projects\ in\ Construction[States]}{max(delivery\ delay, average\ construction\ time)})$$

Once installed, developers eventually decommission projects once the lifetime of the project ends (typically 25 year) but most go through repowering so only a small portion decommission.

$$project\ decommission\ rate[States] = ZIDZ(\frac{Projects\ Installed\ Base[States] * percent\ decommissioned}{average\ project\ lifetime})$$

Repowering often involves larger capacity than the original machines and more modern technology for higher expected annual energy production. This model does not currently capture the aspects of repowering that will become more prevalent as the industry ages. Future work will look at including repowering in a more formal structure.

Both the turbine supply chain the regional wind industry's development capacity affects the project development pipeline. The *total profitable undeveloped projects* (based on the previously discussed breakeven energy production analysis) would directly drive the project development flow except for the limitation on developer capacity. Developers have finite size and resources and have a limit to how fast they can grow. Thus, the relative capacity for project development is much less than the total market potential and grows the longer the exogenous conditions sustain market profitability. A small sub-model (see figure below) for developer capacity dynamics is included and limits the number of projects per year that enter the development pipeline.

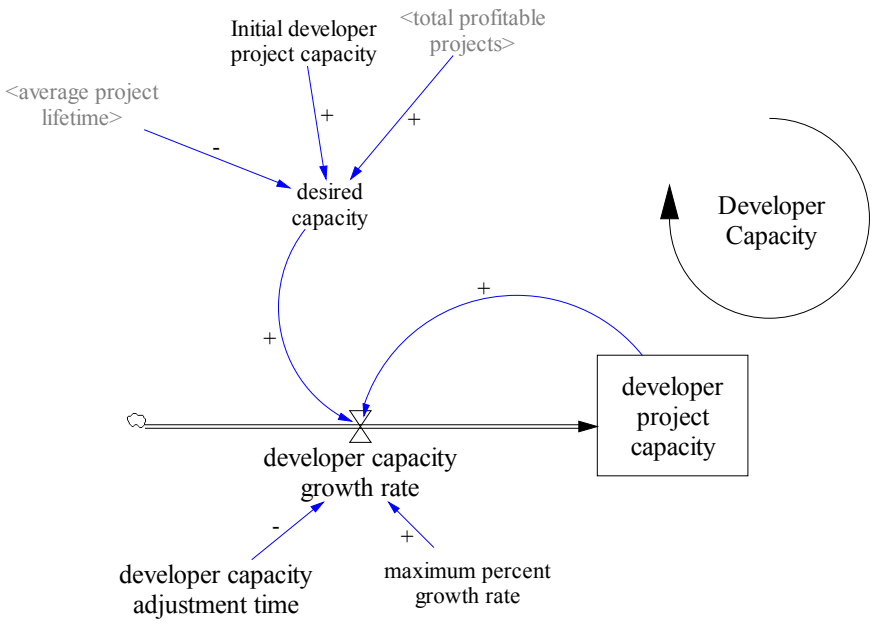

FIGURE 7: DEVELOPER CAPACITY SUB-MODEL

The model hinges on the developer capacity growth rate that is limited from year to year and aims for a desired capacity based on market conditions:

$$developer\ capacity\ growth\ rate[States] = MIN(\frac{desired\ capacity[States] - developer\ project\ capacity[States]}{developer\ capacity\ adjustment\ time}, developer\ project\ capacity[States] * maximum\ percent\ growth\ rate)$$

The maximum growth rate based on analysis of global data is 40% so the current model constraints the growth in development capacity each year by this amount (Eco-Indicators 2010, GWEC, 2014). The desired capacity stems from the total profitable projects over the average project lifetime and has a minimum of the initial developer or "cold-start" developer capacity that is 15 projects based on the early California data (Folkecenter, 1985).

In addition to limits on growth for developer capacity, the turbine supply available for deployment may be less than the market demand. Firms can only build up manufacturing capacity so quickly to match a fast growing market, especially for large equipment as if wind turbines that have unique components that require skilled labor, large facilities, specialized equipment, etc. Prior studies have linked the increasing cost of turbines in 2000s to supply constraints among other factors (Bolinger and Wiser, 2011). During the 1990s, the technology evolved from a relatively small machine to a multi-megawatt machine with blades of the length of 30 m or more. By the end of the 1990s, 1 MW machines were hitting the market. In 2001, due to a series of favorable policies, the market for wind energy machines

grew by 71% and continued to grow the next several years with year-to-year growth rates on average of 35% between 2005 and 2010 (GWEC, 2014). Most production came from a small number of top-tier OEMs and their profits were higher than ever in this seller's market (Bolinger and Wiser, 2011). Turbine costs in 2014 USD rose from under $1000/kW to over $1500/kW or more – a markup of at least 50%. While commodity price increases were a factor in this, so was the tight turbine market supply (Bolinger and Wiser, 2011). In the system dynamics model, we use a standard supply chain model that allows for a wind turbine backlog and an increase in cost of the turbine due to constrained supply. The model bases the turbine supply required on projected installations.

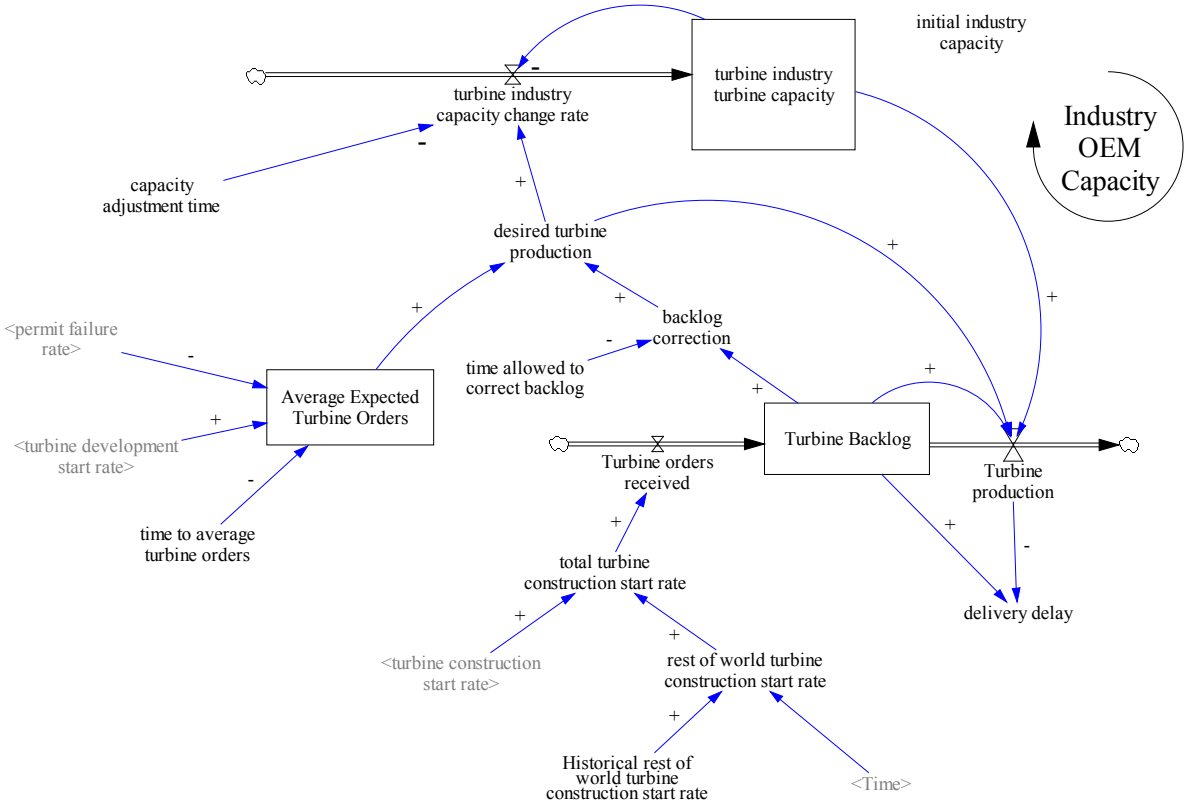

FIGURE 8: TURBINE INDUSTRY CAPACITY AND BACKLOG BASED ON STERMAN, 2000.

The model formulation for the above sub-model is the canonical model from Sterman, 2000. Key equations include the flows of turbine orders received that include the turbines needed for projects going into construction in the states under simulation plus historical turbine demand for the rest of the world for that year. These enter into the turbine backlog and production then is either the full turbine backlog or the industry turbine capacity, whichever is lower:

$$Turbine\ production = MIN(Turbine\ Backlog, MIN(desired\ turbine\ production, turbine\ industry\ turbine\ capacity))$$

As demand outpaces supply, the backlog will grow. The model attempts to correct this by updating the desired turbine production of the industry:

$$desired\ turbine\ production = Average\ Expected\ Turbine\ Orders + backlog\ correction$$

Where the average expected turbine orders adjusts to total projects expected in development:

$$Average\ Expected\ Turbine\ Orders = \left(\frac{SUM(turbine\ development\ start\ rate[States!]) * permit\ failure\ rate - Average\ Expected\ Turbine\ Orders}{time\ to\ average\ turbine\ orders}\right)$$

In the wind industry, while turbine contracts are typically signed at the time when the project goes from development to construction (along with the PPA and financing contracts), sites are designed with specific turbines in mind early in the development process (Splettstosser, 2016). Thus, industry reacts not just to current demand but also expected demand from projects in development. The full sub-model formulation is in the appendix. While turbine industry production capacity is global, the developer capacity is local (as discussed above). Developers build up capability in a region and that capability is not immediately transferable to other regions where the turbine industry can sell its product anywhere.

Co-flows for Land Use, Capacity, and Generation

In addition to the development pipeline, there are also co-flows for turbines, land-use, capacity and electricity generation. In a system dynamics model, a co-flow is one that tracks a primary flow with the same basic structure and flows that depend on those of the primary. The co-flows for turbines, land-use, capacity and electricity generation thus directly mirror the project development pipeline and are in the appendix. Developers have a target capacity in mind at the inception of a project.

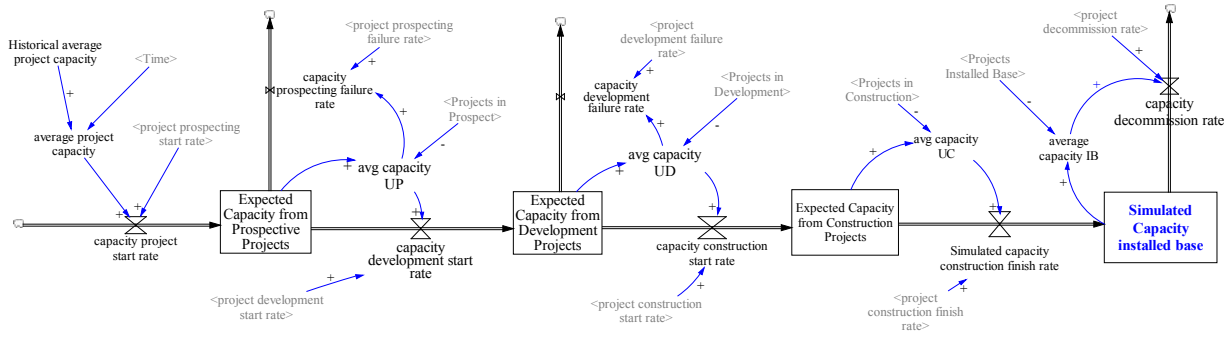

FIGURE 9: CO-FLOW STRUCTURE FOR PROJECT CAPACITY

The co-flow for capacity uses the number of projects going into development:

$$capacity\ project\ start\ rate[States] = Project\ prospecting\ start\ rate[States] * average\ project\ capacity$$

Average project capacity over time for wind projects has fluctuated substantially from very small projects of a few megawatts to large-scale projects of a Gigawatt or more. Average size based on data from the Windpower.net is about 40 MW. Early projects were quite small with 1-7 MW in the late 1970s and early 1980s scaling up to by the mid 1980's (Folkecenter, 1985). With turbines in the size range of 22 to 55 kW, this still meant project sizes of 100 to over 300 turbines. The model uses a lookup curve to allow the project size to grow from 1 MW to 40 MW.

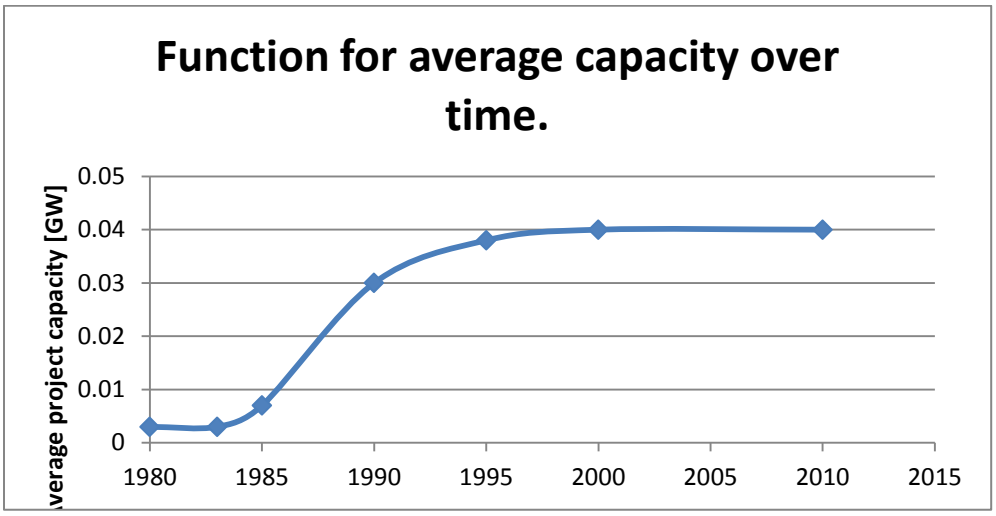

FIGURE 10: FUNCTION FOR AVERAGE CAPACITY OVER TIME. AVERAGE BEGINS AROUND 3 MW FOR EARLY 1980S PROJECTS TO 7 MW BY THE MID-1980S TO 40 MW AS THE CURRENT AVERAGE (THOUGH MANY PROJECTS ARE FAR LARGER; THERE IS A HUGE AMOUNT OF VARIATION IN PROJECT SIZE) (FOLKECENTER, 1985; WINPOWER.NET, 2015).

The turbine co-flow uses the current marginal turbine power rating and the total capacity going into development. Note that the turbine and generation pipelines start at the development stage since developers do not select turbines at the prospecting stage.

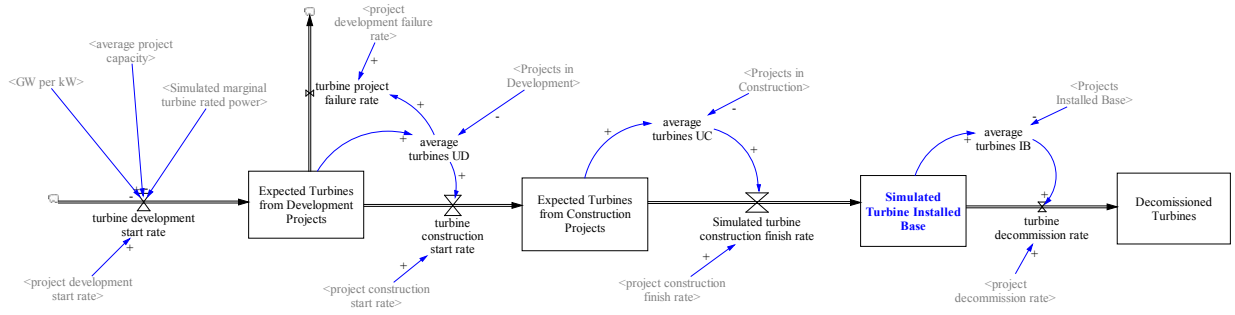

FIGURE 11: TURBINE CO-FLOW STRUCTURE.

$$Simulated\ Turbine\ Development\ Rate[States] = \frac{average\ project\ capacity * Project\ development\ start\ rate[States]}{marginal\ turbine\ power\ rating}$$

The generation co-flow determines the expected generation for the marginal turbine and the project size.

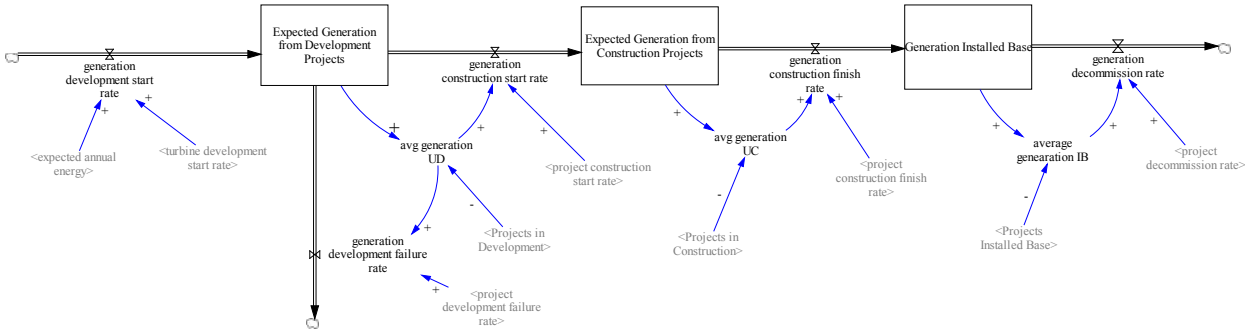

FIGURE 12: GENERATION CO-FLOW STRUCTURE.

$$generation\ development\ start\ rate[States] = expected\ annual\ energy[States] * Simulated\ Turbine\ Development\ Rate[States]$$

The land-use co-flow uses total footprint of each turbine times the number of turbines going into development

$$land\ project\ start\ rate[States] = land\ use\ per\ project * Project\ prospecting\ start\ rate[States]$$

And:

$$land\ use\ per\ project[States] = number\ of\ turbines\ per\ project * area\ per\ turbine$$

And:

$$area\ per\ turbine = (Simulated\ marginal\ rotor\ diameter * Simulated\ marginal\ rotor\ diameter) * spacing\ per\ turbine$$

Spacing per turbine is based on industry heuristics of 7x7-rotor diameter spacing (AWS Scientific, 1997).

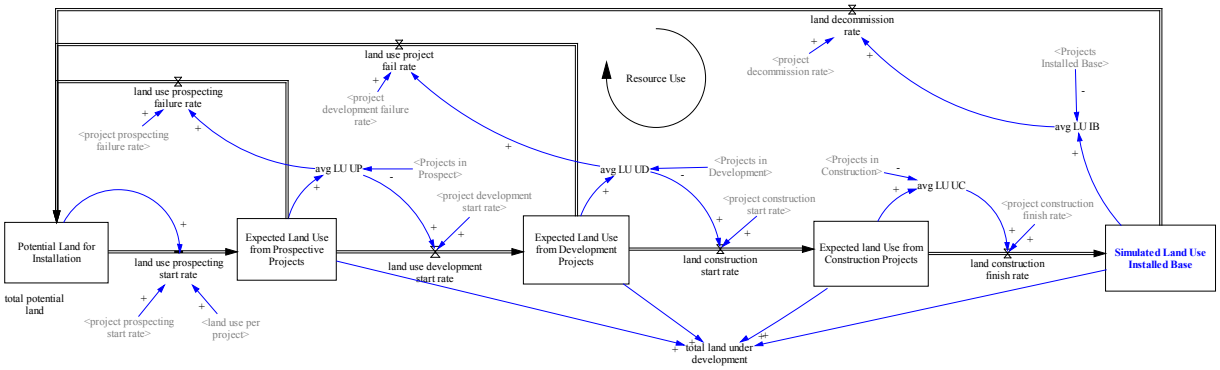

FIGURE 13: LAND-USE CO-FLOW STRUCTURE.

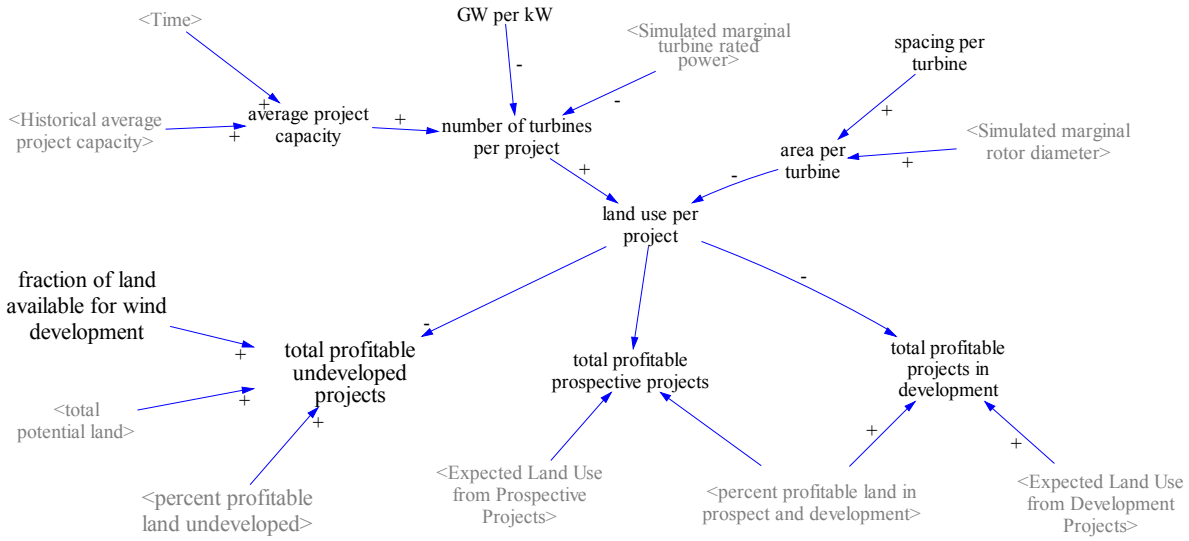

FIGURE 14: MODEL STRUCTURE FOR CALCULATING THE LAND-USE PER PROJECT AS WELL AS THE LAND AVAILABLE FOR DEVELOPMENT THAT IN THE WIND PROJECT SUB-SECTION.

Having described the major elements of the model formulation, the next step involves selection of cases for study, data acquisition, and model calibration. The model is flexible to accommodate a number of different partitions of states grouped as desired. For the purposes of this study, we use the major countries and states for wind development over the last several decades for analysis.

Model Source Code

This appendix contains the full source code of the VENSIM® system dynamics model for the both sets of cases including the four core states for analysis and the larger set used in the second simulation set.

Full model equations

The following list provides the variables, their equations, units and descriptions for the full model set except for those variables that are exogenous inputs to the model developed from the Python script. Those equations follow in the subsequent sections.

---

project prospecting start rate[States]= max(0, MIN(developer project capacity[States]*prospecting sites per developer project, total profitable undeveloped projects[States] * per year))

Units: projects/Year

Description: Prospecting start rate is based on the minimum of either developer project capacity or the total profitable projects.

desired capacity[States]= max(Initial developer project capacity,total profitable projects[States]/average project lifetime)

Units: projects/Year

Description: Desired capacity is the maximum between the initial capacity when no wind market is present and the steady-state develoment capacity of total profitable projects in the region over the average project lifetime.

total profitable projects[States]= total potential land[States]*percent profitable land out of total[States]*fraction of land available for wind development[States]/land use per project

Units: projects

Description: Total profitable projects based on current technology is the total land area of the region multiplied by the percent of the land that is profitable for wind development under current conditions over the amount of land used per project.

project decommission rate[States]=DELAY FIXED(Projects Installed Base[States],average project lifetime,0)

Units: projects / Year

Description: Number of projects decommissioned each year is the total projects in the installed based multiplied by the percent decommissioned delayed by the average project lifetime.

Watt per kW= 1000

Units: dimensionless

Description: Watt to kW converter.

pi= 3.14159

Units: dimensionless

Description: Mathematical constant pi.

turbine development start rate[States]=average project capacity*project development start rate[States]/(Simulated marginal turbine rated power*GW per kW)

Units: turbines / Year

Description: The simulated turbine development rate is the average turbines by project under prospect multiplied byt he project development start rate.

Expected Generation from Construction Projects[States]= INTEG (generation construction start rate[States]-generation construction finish rate[States],0)

Units: kW*hrs

Description: Expected generation from construction projects.

generation construction finish rate[States]= max(0,avg generation UC[States]*project construction finish rate[States])

Units: kW * hrs / Year

Description: Generation construction finish rate is the average generation under construction multiplied by the turbine construction finish rate.

generation development start rate[States]= expected annual energy[States]*turbine development start rate[States]

Units: kW * hrs / Year

Description: Generation development start rate is the expected annual energy for turbine coming onto the market multiplied by the expected number of turbines in development.

average geneaeration IB[States]= ZIDZ(Generation Installed Base[States],Projects Installed Base[States])

Units: kW * hrs / project

Description: Average generation of the installed base is the generation of the installed base over the number of projects (by state).

average capacity IB[States]= ZIDZ(Simulated Capacity installed base[States],Projects Installed Base[States])

Units: GW / project

Description: Average capacity installed base is the capacity by state in the installed based over the number of projects by state in the installed base.

average turbine lifetime= 20

Units: Years

Description: Average turbine lifetime based on NREL Cost and Scaling Model and NREL Wind Cost of Energy Review (Fingersh et al 2006, Mone et al 2015).

capacity project start rate[States]= max(0,project prospecting start rate[States]*average project capacity)

Units: GW / Year

Description: Co-flow for capacity project start rate is the project start rate for projects by the average project capacity.

Simulated capacity construction finish rate[States]= max(0,avg capacity UC[States]*project construction finish rate[States])

Units: GW / Year

Description: Capacity construction finish rate is teh average capacity per project under construction multiplied by the project construction finish rate.

Expected Capacity from Construction Projects[States]= INTEG (capacity construction start rate[States]-Simulated capacity construction finish rate[States],0)

Units: GW

Description: Expected capacity from development projects.

average project capacity= Historical average project capacity(Time)

Units: GW / projects

Description: Average project capacity in GW. 50 MW is typical though projects can range in size from 1 MW to 2 or more GW. Statistics over time give an average value of 40 MW since there are a lot more smaller projects than big projects. This has fluctuated over time so the function is based on a lookup table.

Historical average project capacity([(1980,0)-(2010,0.06)],(1980,0.003),(1983,0.003),(1985,0.007),(1990,0.03),(1995,0.038),(2000,0.04),(2010,0.04))

Units: GW / project

Description: Lookup function for average project capacity factor over time.

Historical IB[States]= Historical Capacity Installed Base[States](Time)

Units: GW

Description: Historical capacity installed base by state to export to results file.

Historical NB[States]= Historical Capacity Construction Rate[States](Time)

Units: GW/Year

Description: Historical new capacity installed base by state to export to results file.

total turbine construction start rate= SUM(turbine construction start rate[States!]) + rest of world turbine construction start rate

Units: turbines / Year

Description: Total turbine construction start rate including states under analysis and the rest of world.

Turbine production= MIN(Turbine Backlog*per year,MIN(desired turbine production,turbine industry turbine capacity))

Units: turbines / Year

Description: Turbine production each year is the minimum between desired turbine production, turbine industry production capacity and turbine backlog (so that the turbine backlog cannot go negative).

rest of world turbine construction start rate= Historical rest of world turbine construction start rate(Time)

Units: turbines / Year

Description: Construction rate for turbines for states outside of the current simulation set.

investment subsidy[States]= Historical investment subsidy[States](Time)

Units: dimensionless

Description: Investment subsidiy per unit power produced from policy support by state.

feed in tariff rate[States]=

Historical feed in tarriff rate[States](Time)

Units: $/(kW*hrs)

Description: FIT incentive per unit energy produced from policy support by state.

delivery delay=

max(0, ZIDZ(Turbine Backlog,Turbine production))

Units: Years

Description: Delivery delay based on current backlog and production rate by taking total backlog over production rate.

rec price[States]= Historical rec price[States](Time)

Units: $/(kW*hrs)

Description: REC incentive per unit energy produced from policy support by state.

production incentive[States]= Historical production incentive[States](Time)

Units: $/(kW*hrs)

Description: Production incentive per unit energy produced from policy support by state.

Turbine Backlog= INTEG (Turbine orders received-Turbine production, 0)

Units: turbines

Description: Turbine backlog due to excess orders relative to turbine industry production capacity where inflows are new orders received and outflows are annual turbine production.

project construction finish rate[States]= ZIDZ(max(Projects in Construction[States],0),max(delivery delay,average construction time))

Units: projects / Year

Description: Projects finishing construction is the total projects under construction over the average construction time for projects.

electricity price[States]= Historical Electricity Prices[States](Time)

Units: $/(kW*hr)

Description: Wholesale electricity price by state in a given year.

backlog effect on turbine price mark([(0,0)-(100000,1)],(0,0),(100,0.1),(500,0.25),(1000,0.3),(10000,0.5),(100000,1))

Units: dimensionless

Description: Look up table for the effect of the size of the wind turbine backlog on the prices seen by the developer. (Look-up table based on a 50% mark-up at 10000 turbines (GE 2008, Bolinger and Wiser, 2011).

initial hub height= 15

Units: meters

Description: Initial hub height of early 1980's wind turbines.

rotor exponent= 2

Units: dimensionless

Description: Rated power scaling exponent based on rotor diameter based on available data.

hub height learning rate= 0.22

Units: dimensionless

Description: Learning rate for turbine hub height based on analysis of available data.

initial power rating= 22

Units: kW / turbine

Description: Initial power rating for early Danish wind turbines of 22 kW

exponent hub height= ln(1 + hub height learning rate) / ln(2)

Units: dimensionless

Description: Exponent for hub height learning rate based on standard formulation.

marginal bos costs= Simulated BOS Costs per unit power*Simulated marginal turbine rated power

Units: $/turbine

Description: Marginal BOS costs taking the BOS costs per unit power multiplied by the marginal turbine power rating.

marginal turbine cost= Simulated marginal turbine rated power*Simulated turbine capital cost*(1+markup on turbine cost due to supply)

Units: $/turbine

Description: Marginal turbine costs taking the turbine costs per unit power multiplied by the marginal turbine power rating.

hub height wind speed max[States]= max wind speed[States]*(Simulated marginal hub height/reference hub height)^wind shear exponent

Units: meters / second

Description: Standard hub height wind speed equation: marginal wind speed at hub height based on hub height relative to reference height scaled by wind shear exponent.

Potential annual energy max[States]= SUM(Cumulative Energy by Wind Speed max[States,windspeed!])

Units: kW*hrs / turbine

Description: Total annual wind energy by summing energy for each wind speed bin.

Rayleigh wind speed probability max[States,windspeed]= ZIDZ(wind speed[windspeed],(hub height wind speed max[States]/SQRT(3.14157/2))^2)*exp(-(ZIDZ(wind speed[windspeed]^2,2*(hub height wind speed max[States]/SQRT(3.14157/2))^2)))

Units: dimensionless

Description: Rayleigh probability by wind speed for site wind resource uses hub height mean wind speed as mean.

expected annual energy max[States]= Potential annual energy max[States]*(1-losses)*Simulated turbine availability

Units: kW * hr / turbine

Description: Total annual energy production by state for the marginal project per turbine by state after losses and turbine downtime are removed.

GW per kW= 1.00E-06

Units: dimensionless

Description: converter from kW units of turbine to GW units for generation and capacity

exponent availability= ln(1-turbine availability learning rate)/ln(2)

Units: dimensionless

Description: Exponent for avaibility learning rate based on standard formulation.

ratio of total to initial turbine installation experience= Simulated total turbine installation experience/initial turbine installation experience

Units: dimensionless

Description: Total installation experience over the initial installation experience.

initial turbine availability= 0.85

Units: dimensionless

Description: Initial turbine availability in the early 1980's

turbine availability learning rate= 0.02

Units: dimensionless

Description: Learning rate for turbine availability based on analysis of available data.

turbine project failure rate[States]= max(0,project development failure rate[States]*average turbines UD[States])

Units: turbines / Year

Description: Co-flow for turbines released each year because they are deemed undevelopable (either due to fluctuating market conditions or particular project weaknesses).

Projects Installed Base[States]= INTEG (project construction finish rate[States]-project decommission rate[States],0)

Units: projects

Description: Installed projects with inflows from projects that have finished construction as well as those that have gone through repowering and outflows for those going into repowering and those that are decommissioned.

avg LU UC[States]= ZIDZ(Expected land Use from Construction Projects[States],Projects in Construction[States])

Units: meters*meters / project

Description: Average land under construction is the total land under construction divided by the number of turbines under construction.

land construction start rate[States]= max(0,avg LU UD[States]*project construction start rate[States])

Units: meters * meters / Year

Description: Land construction start rate is the average land use per turbine under development multiplied by the number of turbines going into construction each year.

land decommission rate[States]= max(0,avg LU IB[States]*project decommission rate[States])

Units: meters * meters / Year

Description: Land that is decommissioned each year derived from the average land use of the installed base by the number of turbines being decommissioned each year.

generation construction start rate[States]= max(0,avg generation UD[States]*project construction start rate[States])

Units: kW * hrs / Year

Description: Generation construction start rate is the average generation under development multiplied by the simulated turbine construction start rate.

generation development failure rate[States]= avg generation UD[States]*project development failure rate[States]

Units: kW * hrs / Year

Description: Co-flow for generation released each year because they are deemed undevelopable (either due to fluctuating market conditions or particular project weaknesses).

avg generation UC[States]= ZIDZ(Expected Generation from Construction Projects[States],Projects in Construction[States])

Units: kW*hrs/project

Description: Average generation under construction is the expected generation under construction over the turbines under construction.

capacity development failure rate[States]= avg capacity UD[States]*project development failure rate[States]

Units: GW / Year

Description: Co-flow for capacity released each year because they are deemed undevelopable (either due to fluctuating market conditions or particular project weaknesses).

land construction finish rate[States]= max(0,avg LU UC[States]*project construction finish rate[States])

Units: meters * meters / Year

Description: Land that is transitioning to the installed base is the average land per turbine under construction multiplied by the number of turbines whose construction is finished each year.

land use project fail rate[States]= max(0,avg LU UD[States]*project development failure rate[States])

Units: meters * meters / Year

Description: Co-flow for land use under development released each year because it is deemed undevelopable (either due to fluctuating market conditions or particular project weaknesses)

capacity prospecting failure rate[States]= project prospecting failure rate[States]*avg capacity UP[States]

Units: GW / Year

Description: Co-flow for capacity released each year because it is deemed undevelopable (either due to fluctuating market conditions or particular project weaknesses).

land use prospecting failure rate[States]= max(0,avg LU UP[States]*project prospecting failure rate[States])

Units: meters * meters / Year

Description: Co-flow for land use under prospect released each year because it is deemed undevelopable (either due to fluctuating market conditions or particular project weaknesses)

project development failure rate[States]= max(0, Projects in Development[States]*percent projects failing[States]/permitting and PPA decision time)

Units: projects / Year

Description: Projects released each year because they are deemed undevelopable (either due to fluctuating market conditions or particular project weaknesses). It is a multiplier of the current projects in development by percent of those projects that fail over the average time it takes for project permitting and signing contracts.

permit failure rate lookup([(1975,0)-(2010,0.8)],(1975,0),(1985,0),(2000,0.75),(2010,0.75))

Units: dimensionless

Description: The permit failure rate is 3 out of 4 projects (Splestosser interview 2016). Projects can fail in either early stage development (from environmental or other permit issues, significant NIMBY issues that surface or failing to secure a power purchase agreement (PPA). It

is assumed that failure rates for projects in the 1970's and early 1980's would have had less of a failure rate due to the market urgency, lack of NIMBYism and generally easier environment for permitting of the time period. Future model: NIMBY issues should come into the project pipeline through affecting this variable.

project construction start rate[States]= max(0, (1-percent projects failing[States])*Projects in Development[States]/permitting and PPA decision time)

Units: projects/Year

Description: Project construction start rate is the percent of projects making it through development (1 - percent of those failing) multiplied by the total projects in development over the average time for project development. This is limited by the number of projects in development deemed profitable (which can change in the time projects move into development due to changing market dynamics.)

project development start rate[States]= max(0,MIN(developer project capacity[States], Projects in Prospect[States]*Percent projects deemed developable[States]))

Units: projects / Year

Description: Project development start rate is the overal projects in development multiplied the percent deemed developable each year.

Percent projects deemed developable[States]= IF THEN ELSE(total profitable prospective projects[States] > 0, 0.8, 0)

Units: 1 / Year

Description: Percent projects deemed developable has been estimated at 1 in 5 or 80% (Splestosser interview 2016). 1 in 5 projects will not be considered beyond prospecting because 20% of land lease rights were not obtained or meteorological information is unfavorable. If market conditions change and all projects are unprofitable, this percentage of success will drop to 0.

permitting and PPA decision time= permitting and PPA decision time lookup(Time*per year)

Units: Years

Description: Project development time is based on a lookup table that changes with time.

project prospecting failure rate[States]= max(0,(1-Percent projects deemed developable[States])*Projects in Prospect[States])

Units: projects/Year

Description: Projects deemed undevelopable released each year because it is deemed undevelopable (either due to fluctuating market conditions or particular project weaknesses)

percent profitable land in prospect and development[States]= max(0,percent profitable land out of total[States]-percent profitable land undeveloped[States]-percent land already built[States])

Units: dimensionless

Description: Percent profitable land in prospect and development is the percent profitable land out of total minus percent already built and that undeveloped.

Turbine orders received= total turbine construction start rate

Units: turbines / Year

Description: Turbine orders received is the total capacity construction start rate divided by the average marginal turbine power rating across all states.

percent land already built[States]= ZIDZ(Expected land Use from Construction Projects[States]+Simulated Land Use Installed Base[States], total potential land[States])

Units: dimensionless

Description: Percent land already built is the percent of land under construction + installed base + decommissioned land over the total potential land.

Expected land Use from Construction Projects[States]= INTEG (land construction start rate[States]-land construction finish rate[States], 0)

Units: meters * meters

Description: Land under construction for wind projects.

time to average turbine orders= 0.25

Units: Years

Description: Delay in averaging turbine order expectations.

desired turbine production= Average Expected Turbine Orders+backlog correction

Units: turbines / Year

Description: Desired turbine production is the sum of turbine orders across all the states and additional production needed to satisfy backlog.

turbine construction start rate[States]= max(0,average turbines UD[States]*project construction start rate[States])

Units: turbines / Year

Description: Turbine construction start rate is the average turbines per project under development multiplied by the project construction start rate.

time allowed to correct backlog= 0.5

Units: Years

Description: Average time allowed to correct any exisiting backlog.

capacity construction start rate[States]= max(0,project construction start rate[States]*avg capacity UD[States])

Units: GW / Year

Description: Capacity construction start rate is the average capacity per project under development multiplied by the project construction start rate.

avg capacity UC[States]= ZIDZ(Expected Capacity from Construction Projects[States],Projects in Construction[States])

Units: GW / project

Description: Average capacity per project under construction is the total capacity under construction divided by the projects in construction.

backlog correction= Turbine Backlog/time allowed to correct backlog

Units: turbines/Year

Description: Backlog correction required in a year is the turbine backlog divided by the time allow for backlog.

Initial developer project capacity= 15

Units: projects/Year

Description: Initial developer capacity based on early project development rates in California.

maximum percent growth rate= 0.4

Units: 1 / Year

Description: Limitation on growth rate for developers by year based on maximum growth in available data.

area per turbine= (Simulated marginal rotor diameter*Simulated marginal rotor diameter)*spacing per turbine

Units: meters * meters / turbine

Description: Area per turbine is the spacing between turbines (7 x 7 or 5 x 50) multiplied by rotor diameter in each direction.

average construction time= 1

Units: Years

Description: Average construction time (Splestosser interview 2016).

average project lifetime= 30

Units: Years

Description: Average project lifetime is estimated at 30 years. Typical project financing is 20 years but projects tend to last 25 years or more (Fingersh, Hand and Laxon, 2006).

per year= 1

Units: 1 / Year

Description: per year for delay functions and in comparison of absolute to per year quantities for capping growth rates

Projects in Construction[States]= INTEG (project construction start rate[States]-project construction finish rate[States],0)

Units: projects

Description: These projects are actively under development: initial resource analysis was favorable and there are efforts to permit the site, negotiate a power purchase agreement, and a turbine supply agreement.

average turbines UC[States]= ZIDZ(Expected Turbines from Construction Projects[States],Projects in Construction[States])

Units: turbines / projects

Description: Average turbines under construction is the expected turbines from construction projects divided by the projects in construction.

average turbines UD[States]= ZIDZ(Expected Turbines from Development Projects[States],Projects in Development[States])

Units: turbines / project

Description: Average turbines under develoment is the total turbines under developent divided by the projects in development.

Expected Turbines from Construction Projects[States]= INTEG (turbine construction start rate[States]-Simulated turbine construction finish rate[States],0)

Units: turbines

Description: Expected turbines from construction projects.

Projects in Prospect[States]= INTEG (project prospecting start rate[States]-project development start rate[States]-project prospecting failure rate[States],0)

Units: projects

Description: These are all the wind projects under consideration by developers in different regions; where met tower data is currently being collected and analyzed. There are inflows of new projects being prospected and outflows for those that transition to development or those that have unfavorable results from prospecting.

Simulated turbine construction finish rate[States]= max(0,average turbines UC[States]*project construction finish rate[States])

Units: turbines / Year

Description: Turbine construction start rate is the average turbiens under construction multiplied by the project construction finish rate.

Projects in Development[States]= INTEG (project development start rate[States]-project development failure rate[States]-project construction start rate[States],0)

Units: projects

Description: Projects in development include inflows from new projects starting development and outflows for projects that failed in development and those that were successful.

prospecting sites per developer project= 5

Units: dimensionless

Description: Ratio of number of sites that get prospect to those that actually get built (5:1 based on developer interviews. Over time has been reduced from 10:1 from better practices and is as low as 3:1 from optimistic estimates) (Splestosser interview 2016).

average turbines IB[States]= ZIDZ(Simulated Turbine Installed Base[States],Projects Installed Base[States])

Units: turbines / projects

Description: Average turbines installed base is the total turbine installed base divided by the total projects installed base.

Projects Decommisioned[States]= INTEG (project decommission rate[States],0)

Units: projects

Description: Projects that are decommissioned due to poor performance - typically older plants that are operating beyond their initial project lifetime.

Rayleigh wind speed probability[States,windspeed]= ZIDZ(wind speed[windspeed],(hub height wind speed[States]/SQRT(pi/2))^2)*exp(-(ZIDZ(wind speed[windspeed]^2,2*(hub height wind speed[States]/SQRT(pi/2))^2)))

Units: dimensionless

Description: Rayleigh probability by wind speed for site wind resource uses hub height mean wind speed as mean.

Power at wind speed[windspeed]= IF THEN ELSE(0.5*air density*Rotor Swept Area*(wind speed[windspeed]^3)*power coefficient/Watt per kW > Simulated marginal turbine rated power, Simulated marginal turbine rated power, 0.5*air density*Rotor Swept Area*(wind speed[windspeed]^3)*power coefficient/Watt per kW)

Units: kW/turbine

Description: Power output for the turbine at a particular wind speed based on standard turbine power formula.

annual maintenance cost[States]= Simulated OPEX per unit energy*expected annual energy[States]

Units: $/ (turbine)

Description: Total annual maintenance costs from maintenance costs per unit energy from the expected annual energy.

Rotor Swept Area= pi*(Simulated marginal rotor diameter/2)^2

Units: meters * meters

Description: Estimated rotor swept area from standard area equation (ignores coning and prebend).

Generation Installed Base[States]= INTEG (generation construction finish rate[States]-generation decommission rate[States],0)

Units: kW * hrs

Description: Generation installed base.

Expected Generation from Development Projects[States]= INTEG (generation development start rate[States]-generation construction start rate[States]-generation development failure rate[States],0)

Units: kW * hrs

Description: Expected generation from development projects.

Expected Turbines from Development Projects[States]= INTEG (turbine development start rate[States]-turbine construction start rate[States]-turbine project failure rate[States],0)

Units: turbines

Description: Expected turbines from development projects.

capacity decommission rate[States]= average capacity IB[States]*project decommission rate[States]

Units: GW / Year

Description: Capacity decommission rate by state.

generation decommission rate[States]= project decommission rate[States]*average genearation IB[States]

Units: kW * hrs / Year

Description: Generation decommission rate based on generation vintage 3 outflow.

Expected Capacity from Prospective Projects[States]= INTEG (capacity project start rate[States]-capacity development start rate[States]-capacity prospecting failure rate[States],0)

Units: GW

Description: Expected capacity from projects under prospect.

annuity factor= (1-(1/(1+interest rate)^(average turbine lifetime*per year)))/(interest rate)

Units: dimensionless

Description: Annuity factor based on NREL Cost and Scaling Model (Fingersh et al 2006).

Simulated Capacity installed base[States]= INTEG (Simulated capacity construction finish rate[States]-capacity decommission rate[States],0)

Units: GW

Description: Total simulated capacity in the installed base by state.

Average Expected Turbine Orders= INTEG ((SUM(turbine development start rate[States!])*permit failure rate-Average Expected Turbine Orders)/time to average turbine orders,100)

Units: turbines / Year

Description: Average turbine orders per year based on average between current year and reference year.

percent projects failing[States]= IF THEN ELSE(total profitable projects in development[States] > 0, permit failure rate, 1)

Units: dimensionless

Description: The percent of projects which will fail is based the permit failure rate unless market conditions change and all projects are unprofitable, then the failure rate increases to 100%.

Total Turbine Construction Completions= SUM(Simulated turbine construction finish rate[States!])+rest of world turbine construction start rate

Units: turbines / Year

Description: Total turbine constructions per year is the sum of finished constructions across all states plus the rest of the world construction finish rate that is not included in the analysis set.

normalized upfront investments[States]= ((marginal bos costs+marginal turbine cost)*(1-investment subsidy[States]))/annuity factor

Units: $/(turbine)

Description: Normalized investment costs from upfront costs after subsidies are removed over the annuity factors.

unit revenue[States]= max(electricity price[States]*cost electricity price discount factor[States]+rec price[States] + production incentive[States], feed in tariff rate[States])

Units: $/ (kW * hr)

Description: Maximum between feed in tarrif rate (if present) and the electricity price reduced by markdown and added to any existing production or REC incentive.

fraction of land available for wind development[States]= 0.2

Units: dimensionless

Description: Fraction of land available for development after fractions from NIMBYism, transmission, pre-developed and other restrictions are removed.

Simulated marginal hub height= initial hub height*(ratio of total to initial turbine installation experience)^exponent hub height

Units: meters

Description: Simulated marginal hub height scaled from the initial hub height scaled by the total experience to initial experience to the learning exponent.

Simulated Land Use Installed Base[States]= INTEG (land construction finish rate[States]-land decommission rate[States],0)

Units: meters * meters

Description: Total land occupied by turbine projects (includes space between turbines).

Simulated marginal turbine rated power= initial power rating*(Simulated marginal rotor diameter/initial rotor size)^rotor exponent

Units: kW / turbine

Description: Scaling of power rating over time based on empirical relationship initial rotor diameter by power rating scaled by the rotor power exponent.

upfront investment costs[States]= marginal turbine cost+marginal bos costs

Units: $/ turbine

Description: Upfront investment costs per turbine summing the turbine and balance of station costs per unit power multiplied turbine power rating.

annual breakeven costs[States]= annual maintenance cost[States]*(1-tax rate)+normalized upfront investments[States]

Units: $/ (turbine)

Description: Annual breakeven costs are the annual operations costs after tax exemptions added to normalized upfront investment costs.

Simulated marginal LCOE[States]= ZIDZ(upfront investment costs[States]*fixed charge rate+ annual maintenance cost[States]*(1-tax rate), expected annual energy[States])

Units: $/(kW*hr)

Description: Simplified NREL Cost and Scaling Model LCOE calculation using fixed charge rate for financing, upfront investment costs and operational expenditures minus tax exemptions over expected annual energy.

markup on turbine cost due to supply= backlog effect on turbine price mark(Turbine Backlog)

Units: dimensionless

Description: Price mark-up on turbines due to supply constraints based on lookup function and the size of the current wind turbine backlog.

land use prospecting start rate[States]= IF THEN ELSE(Potential Land for Installation[States]*per year - land use per project*max(0,project prospecting start rate[States]) >=0, land use per project*max(0,project prospecting start rate[States]), max(0, Potential Land for Installation[States]*per year))

Units: meters*meters/Years

Description: Transition rate from land that has never been prospected or developed for a wind project to one that is under prospecting. Derived from rate of projects beginning prospecting each year and the average land use per project. If all avilable land has already been considered for development, the value will be 0.

total profitable prospective projects[States]= Expected Land Use from Prospective Projects[States]/land use per project*percent profitable land in prospect and development[States]

Units: projects

Description: Projects that are profitable in prospect are those that have already begun prospecting and are profitable under current market conditions.

total profitable undeveloped projects[States]= total potential land[States]*fraction of land available for wind development[States]*percent profitable land undeveloped[States]/land use per project

Units: projects

Description: Total profitable projects is based on the percent of undeveloped land that is currently considered profitable multiplied by the total potential land available for wind development divided by the land use per project.

Cumulative Energy by Wind Speed max[States,windspeed]= Power at wind speed[windspeed]*Rayleigh wind speed probability max[States,windspeed]*hours per year

Units: kW*hrs / turbine

Description: Cumulative energy by wind speed based on Rayleigh probability and power output at that wind speed.

land use per project= number of turbines per project*area per turbine

Units: meters * meters / project

Description: Land use per project is the average project capacity divided by the capacity per turbine and multiplied by the are per turbine.

number of turbines per project= average project capacity/(GW per kW*Simulated marginal turbine rated power)

Units: turbines / project

Description: Turbines per project are determined by the capacity of the project over the current marginal turbine rated power.

Cumulative Energy by Wind Speed[States,windspeed]= Power at wind speed[windspeed]*Rayleigh wind speed probability[States,windspeed]*hours per year

Units: kW*hrs / turbine

Description: Cumulative energy by wind speed based on Rayleigh probability and power output at that wind speed.

total profitable projects in development[States]= Expected Land Use from Development Projects[States]/land use per project*percent profitable land in prospect and development[States]

Units: projects

Description: Projects that are profitable in development are those that have already begun prospecting and are profitable under current market conditions.

percent profitable land out of total[States]= IF THEN ELSE(breakeven expected annual energy[States]>expected annual energy max[States],0,MIN(1,ABS(ZIDZ((expected annual energy max[States]-breakeven expected annual energy[States]),linear term for wind speed by land use[States])+ZIDZ((expected annual energy max[States]-breakeven expected annual energy[States])^2,quadratic term for wind speed by land use[States]))))

Units: dimensionless

Description: Percent profitable land based on breakeven annual energy and expected annual energy characteristics.

Simulated marginal rotor diameter= initial rotor size*(ratio of total to initial turbine installation experience)^exponent rotor diameter

Units: meters

Description: Simulated marginal rotor diameter scaled from the initial rotor diameter scaled by the total experience to initial experience to the learning exponent.

Simulated OPEX per unit energy= initial annual OPEX costs per unit energy*(ratio of total to initial turbine installation experience)^exponent OPEX

Units: $/ (kW * hrs)

Description: Average overall O&M costs based for land lease, standard O&M and component replacements.

Simulated turbine capital cost= initial upfront turbine costs per unit power*(ratio of total to initial turbine installation experience)^exponent turbine costs

Units: $/ kW

Description: Turbine costs per unit power over time based on initial cost estimate per unit power scaled by total to initial experience scaled by the cost exponent for the turbine cost learning rate.

initial annual OPEX costs per unit energy= 0.05

Units: $/(kW * hrs)

Description: Normal O&M costs for standard maintenance activities, land leases, etc.

Simulated turbine availability= initial turbine availability*(1/((ratio of total to initial turbine installation experience)^exponent availability))

Units: dimensionless

Description: Average turbine availability is the 1 minus the percent of turbine downtime (calculated by the average time lost per breakdown by total turbine breakdowns per year over the total time in a year).

Simulated BOS Costs per unit power= initial balance of station costs per unit power*(ratio of total to initial turbine installation experience)^exponent bos costs

Units: $/kW

Description: Balance of station costs per unit power over time based on initial cost estimate per unit power scaled by total to initial experience scaled by the cost exponent for the balance of station learning rate.

Potential Land for Installation[States]= INTEG (land decommission rate[States]+land use prospecting failure rate[States]+land use project fail rate[States]-land use prospecting start rate[States],total

potential land[States]-Simulated Land Use Installed Base[States]-Expected Land Use from Development Projects[States]-Expected Land Use from Prospective Projects[States])

Units: meters * meters

Description: Potential land available for installations is all land that has never been prospected or begun development for wind energy (includes undevelopable land such as urban areas, environmentally sensitive areas, etc.)

Expected Land Use from Prospective Projects[States]= INTEG (land use prospecting start rate[States]-land use development start rate[States]-land use prospecting failure rate[States],0)

Units: meters*meters

Description: Land under prospect for wind projects.

Expected Capacity from Development Projects[States]= INTEG (capacity development start rate[States]-capacity construction start rate[States]-capacity development failure rate[States],0)

Units: GW

Description: Expected capacity from development projects.

avg generation UD[States]= ZIDZ(Expected Generation from Development Projects[States],Projects in Development[States])

Units: kW * hrs / project

Description: Average generation under development is the expected generation from development over the turbines from development.

land use development start rate[States]= max(0,avg LU UP[States]*project development start rate[States])

Units: meters*meters / Year

Description: Transition rate from land being prospected to land being developed based on average land use per turbine being prospected multiplied by the number of turbines transitioning from prospect to development.

avg LU IB[States]= ZIDZ(Simulated Land Use Installed Base[States],Projects Installed Base[States])

Units: (meters * meters) / project

Description: Average land use of installed base based on the total land of the install based and the number of turbines installed.

avg LU UD[States]= ZIDZ(Expected Land Use from Development Projects[States],Projects in Development[States])

Units: meters*meters / project

Description: Average land use under development is the land under development divided the number of turbines under development.

avg LU UP[States]= ZIDZ(Expected Land Use from Prospective Projects[States],Projects in Prospect[States])

Units: meters*meters / project

Description: Average land use per project under prospecting is derived from the current expected land use over the number of turbines in prospective projects.

Expected Land Use from Development Projects[States]= INTEG (land use development start rate[States]-land construction start rate[States]-land use project fail rate[States],0)

Units: meters*meters

Description: Land under development for wind projects.

permit failure rate= permit failure rate lookup(Time*per year)

Units: dimensionless [0,1,0.05]

Description: Permit failure rate is based on a lookup table that changes with time.

total land under development[States]= Expected Land Use from Prospective Projects[States]+Expected Land Use from Development Projects[States]+Expected land Use from Construction Projects[States]+Simulated Land Use Installed Base[States]

Units: meters * meters

Description: Total land under development is all land under prospect, development, construction, installed and land that has been decommissioned.

capacity development start rate[States]= max(0,avg capacity UP[States]*project development start rate[States])

Units: GW / Year

Description: Capacity development start rate is the average capacity per project under prospect multiplied by the project development start rate.

turbine industry turbine capacity= INTEG (turbine industry capacity change rate, initial industry capacity)

Units: turbines/Year

Description: Turbine industry annual production capacity.

avg capacity UP[States]= ZIDZ(Expected Capacity from Prospective Projects[States],Projects in Prospect[States])

Units: GW / project

Description: Average capacity per project under prospect based on the total capacity from prospective projects divided by the project under prospect.

avg capacity UD[States]= ZIDZ(Expected Capacity from Development Projects[States],Projects in Development[States])

Units: GW / project

Description: Average capacity per project under development is the total capacity under development divided by the number of projects in development.

turbine industry capacity change rate= (desired turbine production-turbine industry turbine capacity)/capacity adjustment time

Units: turbines / (Year * Year)

Description: Change in turbine industry production capacity each year is the difference in desired capacity to current capacity over the adjustment time.

developer project capacity[States]= INTEG (developer capacity growth rate[States],Initial developer project capacity)

Units: projects/Year

Description: Developer project development capacity by state which asjusts up or down (down if the develoepr capacity growth rate is negative).

developer capacity growth rate[States]=MIN((desired capacity[States]-developer project capacity[States])/developer capacity adjustment time, developer project capacity[States]*maximum percent growth rate)

Units: projects / (Year * Year)

Description: Developer capacity by state adjusting to market demand by state.

Simulated Turbine Installed Base[States]= INTEG (Simulated turbine construction finish rate[States]-turbine decommission rate[States],0)

Units: turbines

Description: Total simulated turbine installed base by state.

turbine decommission rate[States]= max(0,average turbines IB[States]*project decommission rate[States])

Units: turbines / Year

Description: Turbine decomission rate is the average turbines in the installed base multiplied by the project decommission rate.

interest rate= 0.08

Units: dimensionless

Description: Interest rate based on NREL Cost and Scaling Model (Fingersh et al 2006).

marginal install wind speed[States]= max(0,max wind speed[States]+linear term for wind speed by land use[States]*(percent land under development[States])+quadratic term for wind speed by land use[States]*(percent land under development[States])^2)

Units: meters / second

Description: Marginal mean wind speed based on look-up function by state based on GIS analysis.

percent land under development[States]= ZIDZ(total land under development[States],total potential land[States])

Units: dimensionless

Description: Percent land under development is total land under development by total potential land.

percent profitable land undeveloped[States]=max(percent profitable land out of total[States]-percent land under development[States],0)

Units: dimensionless

Description: Percent profitable land available for development is the percent of total profitable land minus that already under development.

breakeven expected annual energy[States]= ZIDZ(annual breakeven costs[States],unit revenue[States])

Units: kW * hrs / turbine

Description: Breakeven annual energy level required based on current costs divided by current revenue.

Potential annual energy[States]= SUM(Cumulative Energy by Wind Speed[States,windspeed!])

Units: kW*hrs / turbine

Description: Total annual wind energy by summing energy for each wind speed bin.

expected annual energy[States]= Potential annual energy[States]*(1-losses)*Simulated turbine availability

Units: kW * hr / turbine

Description: Total annual energy production by state for the marginal project per turbine by state after losses and turbine downtime are removed.

windspeed: w1, w2, w3, w4, w5, w6, w7, w8, w9, w10, w11, w12, w13, w14, w15, w16, w17, w18, w19, w20, w21, w22, w23, w24, w25

Units:

Description: wind speed subscript

air density= 1.225

Units: dimensionless

Description: Standard air density constant is 1.225 kg / m^3 at sea level at 15 degrees Celsius (AWS Scientific, 1997).

power coefficient= 0.43

Units: dimensionless

Description: The model uses a power coefficient that multiplies a reasonable rotor power coefficient of 0.47 by a reasonable drivetrain efficiency of 0.92 for a total of 0.43 (Fingersh, Hand and Laxon, 2006)

wind speed[w1]= 1 ~~|

wind speed[w2]= 2 ~~|

wind speed[w3]= 3 ~~|

wind speed[w4]= 4 ~~|

wind speed[w5]= 5 ~~|

wind speed[w6]= 6 ~~|

wind speed[w7]= 7 ~~|

wind speed[w8]= 8 ~~|

wind speed[w9]= 9 ~~|

wind speed[w10]= 10 ~~|

wind speed[w11]= 11 ~~|

wind speed[w12]= 12 ~~|

wind speed[w13]= 13 ~~|

wind speed[w14]= 14 ~~|

wind speed[w15]= 15 ~~|

wind speed[w16]= 16 ~~|

wind speed[w17]= 17 ~~|

wind speed[w18]= 18 ~~|

wind speed[w19]= 19 ~~|

wind speed[w20]= 20 ~~|

wind speed[w21]= 21 ~~|

wind speed[w22]= 22 ~~|

wind speed[w23]= 23 ~~|

wind speed[w24]= 24 ~~|

wind speed[w25]= 25

Units: meters / second

Description: wind speed units constant for power curve calcuations.

hub height wind speed[States]= marginal install wind speed[States]*(Simulated marginal hub height/reference hub height)^wind shear exponent

Units: meters / second

Description: Standard hub height wind speed equation: marginal wind speed at hub height based on hub height relative to reference height scaled by wind shear exponent.

reference hub height=50

Units: meters

Description: Reference hub height where wind speed data is measured.

wind shear exponent= 0.143

Units: dimensionless

Description: Standard wind shear exponent for flat terrain is 1/7 which equals 0.143(AWS Scientific, 1997).

developer capacity adjustment time= 1

Units: Year

Description: Time it takes a developer to adjust capacity to desired capacity.

initial balance of station costs per unit power= 1000

Units: $/kW

Description: Initial balance of station costs per unit power.

turbine bos learning rate= 0.073

Units: dimensionless

Description: Learning rate for turbine bos costs based on analysis of available data.

exponent bos costs= ln(1-turbine bos learning rate)/ln(2)

Units: dimensionless

Description: Exponent for BOS costs learning using standard formulation.

tax rate= 0.4

Units: dimensionless

Description: Corporate tax exemption rate on operational expenditures based on NREL Cost and Scaling Model (Fingersh et al 2006).

losses= 0.15

Units: dimensionless

Description: Typical plant losses for an assortment of factors.

fixed charge rate= 0.12

Units: dimensionless

Description: Financing fixed charge rate based on NREL Cost and Scaling Model and NREL Wind Cost of Energy Review (Fingersh et al 2006, Mone et al 2015).

data: turbinesnew, turbinesbase, turbinestotal, capacitynew, capacitybase, capacitytotal, rating, rotordiameter, hubheight, turbinecosts, boscosts, opex, lcoe

Units:

Description: Data subscripts

exponent OPEX= ln(1-turbine OPEX learning rate)/ln(2)

Units: dimensionless

Description: Exponent for reliability based on learning rate using standard learning rate formulation.

turbine OPEX learning rate= 0.073

Units: dimensionless

Description: Learning rate for turbine operational expenditures based on analysis of available data.

exponent turbine costs= ln(1-turbine cost learning rate)/ln(2)

Units: dimensionless

Description: Exponent for turbine costs learning using standard formulation.

turbine cost learning rate= 0.073

Units: dimensionless

Description: Learning rate for turbine costs based on analysis of available data.

Simulated total turbine installation experience= INTEG (Total Turbine Construction Completions-installation experience loss rate, initial turbine installation experience)

Units: turbines

Description: Simulated total turbine installation experience.

Decomissioned Turbines[States]= INTEG (turbine decommission rate[States],0)

Units: turbines

Description: Total decommissioned turbines.

rotor diameter learning rate= 0.25

Units: dimensionless

Description: Learning rate for turbine rotor diameter based on analysis of available data.

exponent rotor diameter= ln(1 + rotor diameter learning rate) / ln(2)

Units: dimensionless

Description: Exponent for rotor diameter learning rate based on standard formulation.

initial rotor size= 10

Units: meters

Description: Initial rotor size for turbines installed in the 1970s.

hours per year= 24*365

Units: hr

Description: Hours per year are 8760 (365 days * 24 hours per day)

capacity adjustment time= 0.25

Units: Years

Description: Time for turbine industry to adjust capacity up/down to target.

cost electricity price discount factor[States]= 0.6

Units: dimensionless

Description: Discount factor from wholesale or consumer electricity price to cost (i.e. to remove utility profit mark-up, and any embedded transmission or distribution costs).

initial upfront turbine costs per unit power= 2000

Units: $/kW

Description: Initial turbine costs per unit power.

permitting and PPA decision time lookup([(1975,1)-(2010,4)],(1975,1),(1985,1),(1990,4),(2010,4))

Units: Years [0,2,0.02]

Description: Project development time (including permitting and contracting) is typically 5 years including 1 year for prospecting (4 years in exclusion of prospecting time) for current projects (Splestosser interview 2016). It is assumed that early projects in the 70's or 80's were much easier to permit for a number of reasons including market urgency, lack of NIMBYism, and generally less complex permitting requirements during the time period.

initial industry capacity= 100

Units: turbines / Year [0,1e+008]

Description: Initial turbine industry capacity.

initial turbine installation experience= 300

Units: turbines

Description: 300 turbines installed prior to 1980 in unknown years and is used as a starting point in the turbine technology and learning curves.

installation experience loss rate= Simulated total turbine installation experience*experience loss rate

Units: turbines / Year

Description: Experience loss rate is based on the total installation experience multiplied by the experience / knowledge loss rate.

experience loss rate= 0

Units: 1 / Years

Description: Experience loss rate is the loss of knowledge experiene per year. It is currently set to zero since learning rates are statistically estimated from cumulative installations.

spacing per turbine= 50

Units: 1 / turbine

Description: Spacing per turbine is assumed to be a standard 7 x 7 rotor diameter spacing (or 5 x 10 for farms with more preferential directions) (AWS Scientific, 1997).

Simulated[turbinesbase,States]= Simulated Turbine Installed Base[States] ~~|

Simulated[turbinesnew,States]=Simulated turbine construction finish rate[States] ~~|

Simulated[turbinestotal, States]=SUM(Simulated Turbine Installed Base[States!]) ~~|

Simulated[capacitybase,States]=Simulated Capacity installed base[States] ~~|

Simulated[capacitynew, States]= Simulated capacity construction finish rate[States] ~~|

Simulated[rating, States]= Simulated marginal turbine rated power ~~|

Simulated[rotordiameter, States]= Simulated marginal rotor diameter ~~|

Simulated[lcoe, States]= Simulated marginal LCOE[States] ~~|

Simulated[turbinecosts, States]= Simulated turbine capital cost ~~|

Simulated[boscosts,States]= Simulated BOS Costs per unit power ~~|

Simulated[opex,States]= Simulated OPEX per unit energy ~~|

Simulated[hubheight,States]= Simulated marginal hub height ~~|

Simulated[capacitytotal,States]= SUM(Simulated Capacity installed base[States!])

Units: Units

Description: Set equal to the simulated data series.

Historical[turbinesbase,States]= Historical Turbine Installed Base[States](Time) ~~|

Historical[turbinesnew,States]= Historical Turbine Construction Rate[States](Time) ~~|

Historical[turbinestotal,States]= SUM(Historical Turbine Installed Base[States!](Time)) ~~|

Historical[capacitybase, States]=Historical Capacity Installed Base[States](Time) ~~|

Historical[capacitynew, States]= Historical Capacity Construction Rate[States](Time) ~~|

Historical[rating, States]=Historical Marginal Rated Power(Time) ~~|

Historical[rotordiameter,States]=Historical Marginal Rotor Diameter(Time) ~~|

Historical[lcoe,States]= Historical Marginal LCOE(Time) ~~|

Historical[turbinecosts, States]= Historical Turbine Cost(Time) ~~|

Historical[boscosts,States]=Historical BOS Costs(Time) ~~|

Historical[opex,States]= Historical OPEX Costs(Time) ~~|

Historical[hubheight,States]=Historical Marginal Hub Height(Time) ~~|

Historical[capacitytotal,States]= SUM(Historical Capacity Installed Base[States!](Time))

Units: Units

Description: Set equal to the historical data series.

****

.Control

****

FINAL TIME = 2015

Units: Year

Description: The final time for the simulation.

INITIAL TIME = 1980

Units: Year

Description: The initial time for the simulation.

SAVEPER = TIME STEP

Units: Year [0,?]

Description: The frequency with which output is stored.

TIME STEP = 0.25

Units: Year [0,?]

Description: The time step for the simulation.

First model set: historically dominant wind markets

States: Denmark, Germany, Spain, California

Units:

Description: States in the analysis.

Statescopy: Denmark, Germany, Spain, California

Units:

Description: Copy of States in the analysis for simulation.

total potential land[Denmark]= 4.11035e+010 ~~|

total potential land[Germany]= 3.55246e+011 ~~|

total potential land[Spain]= 5.0325e+011 ~~|

total potential land[California]= 4.03466e+011

Units: meters * meters

Description: Total available land area in a given state.

max wind speed[Denmark]= 7.17 ~~|

max wind speed[Germany]= 9.47 ~~|

max wind speed[Spain]= 7.63 ~~|

max wind speed[California]= 6.46

Units: meters / second

Description: Maximum wind speed in a given region.

linear term for wind speed by land use[Denmark]= -2.9574 ~~|

linear term for wind speed by land use[Germany]= -5.65748 ~~|

linear term for wind speed by land use[Spain]= -5.2201 ~~|

linear term for wind speed by land use[California]= -1.97194

Units: meters / second

Description: Linear coefficient for polynomial fit of wind speed resource by land use.

quadratic term for wind speed by land use[Denmark]= -0.63987 ~~|

quadratic term for wind speed by land use[Germany]= 2.58684 ~~|

quadratic term for wind speed by land use[Spain]= 1.39668 ~~|

quadratic term for wind speed by land use[California]= 0.64225

Units: meters / second

Description: Quadratic coefficient for polynomial fit of wind speed resource by land use.

Historical Electricity Prices[Denmark]([(0,0)-(2014,10)], (1980, 0.022629), (1981, 0.037994), (1982, 0.052032), (1983, 0.058624), (1984, 0.057648), (1985, 0.070801), (1986, 0.068542), (1987, 0.060822), (1988, 0.069275), (1989, 0.079102), (1990, 0.08197), (1991, 0.07959), (1992, 0.082092), (1993, 0.076599), (1994, 0.075562), (1995, 0.075195), (1996, 0.078491), (1997, 0.078125), (1998, 0.082336), (1999, 0.083557), (2000, 0.086731), (2001, 0.093933), (2002, 0.1001), (2003, 0.10645), (2004, 0.10724), (2005, 0.10968), (2006, 0.12097), (2007, 0.12769), (2008, 0.14734), (2009, 0.13843), (2010, 0.13953), (2011, 0.13855), (2012, 0.13928), (2013, 0.13611), (2014, 0.13135)) ~~|

Historical Electricity Prices[Germany]([(0,0)-(2014,10)], (1980, 0.12292), (1981, 0.12207), (1982, 0.12164), (1983, 0.11945), (1984, 0.13184), (1985, 0.13464), (1986, 0.13403), (1987, 0.13879), (1988, 0.1355), (1989, 0.13342), (1990, 0.14209), (1991, 0.14819), (1992, 0.15442), (1993, 0.15417), (1994, 0.14807), (1995, 0.13367), (1996, 0.13477), (1997, 0.13757), (1998, 0.13623), (1999, 0.13525), (2000, 0.13379), (2001, 0.13831), (2002, 0.13538), (2003, 0.13989), (2004, 0.14063), (2005, 0.14209), (2006, 0.13928), (2007, 0.1322), (2008, 0.13025), (2009, 0.13147), (2010, 0.11963), (2011, 0.11957), (2012, 0.12042), (2013, 0.11859), (2014, 0.11597)) ~~|

Historical Electricity Prices[Spain]([(0,0)-(2014,10)], (1980, 0.018494), (1981, 0.018494), (1982, 0.054535), (1983, 0.070618), (1984, 0.068054), (1985, 0.093079), (1986, 0.091248), (1987, 0.092163), (1988, 0.10132), (1989, 0.1153), (1990, 0.11987), (1991, 0.13879), (1992, 0.14441), (1993, 0.13416), (1994, 0.12891), (1995, 0.13025), (1996, 0.1322), (1997, 0.12793), (1998, 0.11591), (1999, 0.11328), (2000, 0.10828), (2001, 0.10107), (2002, 0.10071), (2003, 0.10217), (2004, 0.10406), (2005, 0.10449), (2006, 0.11041), (2007, 0.12225), (2008, 0.14001), (2009, 0.15686), (2010, 0.17139), (2011, 0.17017), (2012, 0.17126), (2013, 0.16736), (2014, 0.16138)) ~~|

Historical Electricity Prices[California]([(0,0)-(2014,10)], (1980, 0.063293), (1981, 0.072449), (1982, 0.083618), (1983, 0.084229), (1984, 0.088501), (1985, 0.095093), (1986, 0.092834), (1987, 0.088745), (1988, 0.094849), (1989, 0.10297), (1990, 0.10437), (1991, 0.11316), (1992, 0.11591), (1993, 0.11743), (1994, 0.1189), (1995, 0.12079), (1996, 0.11761), (1997, 0.11951), (1998, 0.10779), (1999, 0.10919), (2000, 0.11981), (2001, 0.1311), (2002, 0.13782), (2003, 0.13306), (2004, 0.13208), (2005, 0.13477), (2006, 0.15552), (2007, 0.15466), (2008, 0.15222), (2009, 0.16223), (2010, 0.15698), (2011, 0.15784), (2012, 0.16138), (2013, 0.11578), (2014, 0.11578))

Units: $/ (kW * hrs)

Description: Lookup function for electricity price based on year.

Historical production incentive[Denmark]([(0,0)-(2014,10)], (1980, 0.044006), (1981, 0.044006), (1982, 0.044006), (1983, 0.044006), (1984, 0.044006), (1985, 0.044006), (1986, 0.044006), (1987, 0.044006), (1988, 0.044006), (1989, 0.044006), (1990, 0.044006), (1991, 0.044006), (1992, 0.044006), (1993, 0.044006), (1994, 0.044006), (1995, 0.044006), (1996, 0.044006), (1997, 0.044006), (1998, 0.044006), (1999, 0.044006), (2000, 0.044006), (2001, 0.044006), (2002, 0.044006), (2003, 0.044006), (2004, 0.044006), (2005, 0.044006), (2006, 0.044006), (2007, 0.044006), (2008, 0.044006), (2009, 0.044006), (2010, 0.044006), (2011, 0.044006), (2012, 0.044006), (2013, 0.044006), (2014, 0.044006)) ~~|

Historical production incentive[Germany]([(0,0)-(2014,10)], (1980, 0), (1981, 0), (1982, 0), (1983, 0), (1984, 0), (1985, 0), (1986, 0), (1987, 0), (1988, 0), (1989, 0), (1990, 0), (1991, 0), (1992, 0), (1993, 0), (1994, 0), (1995, 0), (1996, 0), (1997, 0), (1998, 0), (1999, 0), (2000, 0), (2001, 0), (2002, 0), (2003, 0), (2004, 0), (2005, 0), (2006, 0), (2007, 0), (2008, 0), (2009, 0), (2010, 0), (2011, 0), (2012, 0), (2013, 0), (2014, 0)) ~~|

Historical production incentive[Spain]([(0,0)-(2014,10)], (1980, 0), (1981, 0), (1982, 0), (1983, 0), (1984, 0), (1985, 0), (1986, 0), (1987, 0), (1988, 0), (1989, 0), (1990, 0), (1991, 0), (1992, 0), (1993, 0), (1994, 0), (1995, 0), (1996, 0), (1997, 0), (1998, 0), (1999, 0), (2000, 0), (2001, 0), (2002, 0), (2003, 0), (2004, 0), (2005, 0), (2006, 0), (2007, 0), (2008, 0), (2009, 0), (2010, 0), (2011, 0), (2012, 0), (2013, 0), (2014, 0)) ~~|

Historical production incentive[California]([(0,0)-(2014,10)], (1980, 0), (1981, 0), (1982, 0), (1983, 0), (1984, 0), (1985, 0), (1986, 0), (1987, 0), (1988, 0), (1989, 0), (1990, 0), (1991, 0), (1992, 0.032074), (1993, 0.031128), (1994, 0.03035), (1995, 0.02951), (1996, 0.028671), (1997, 0.02803), (1998, 0.027603), (1999, 0.026993), (2000, 0), (2001, 0.025391), (2002, 0), (2003, 0.024445), (2004, 0), (2005, 0.023026), (2006, 0.022308), (2007, 0.021698), (2008, 0.020889), (2009, 0.022995), (2010, 0.022995), (2011, 0.022995), (2012, 0.022995), (2013, 0.022995), (2014, 0.022995))

Units: $/ (kW * hrs)

Description: Lookup function for production incentive based on year.

Historical investment subsidy[Denmark]([(0,0)-(2014,10)], (1980, 0.3999), (1981, 0.3999), (1982, 0.3999), (1983, 0.3999), (1984, 0.3999), (1985, 0.3999), (1986, 0.3999), (1987, 0.3999), (1988, 0.3999), (1989, 0.3999), (1990, 0), (1991, 0), (1992, 0), (1993, 0), (1994, 0), (1995, 0), (1996, 0), (1997, 0), (1998, 0), (1999, 0), (2000, 0), (2001, 0), (2002, 0), (2003, 0), (2004, 0), (2005, 0), (2006, 0), (2007, 0), (2008, 0), (2009, 0), (2010, 0), (2011, 0), (2012, 0), (2013, 0), (2014, 0)) ~~|

Historical investment subsidy[Germany]([(0,0)-(2014,10)], (1980, 0), (1981, 0), (1982, 0), (1983, 0), (1984, 0), (1985, 0), (1986, 0), (1987, 0), (1988, 0), (1989, 0), (1990, 0), (1991, 0), (1992, 0), (1993, 0), (1994, 0), (1995, 0), (1996, 0), (1997, 0), (1998, 0), (1999, 0), (2000, 0), (2001, 0), (2002, 0), (2003, 0), (2004, 0), (2005, 0), (2006, 0), (2007, 0), (2008, 0), (2009, 0), (2010, 0), (2011, 0), (2012, 0), (2013, 0), (2014, 0)) ~~|

Historical investment subsidy[Spain]([(0,0)-(2014,10)], (1980, 0), (1981, 0), (1982, 0), (1983, 0), (1984, 0), (1985, 0), (1986, 0), (1987, 0), (1988, 0), (1989, 0), (1990, 0), (1991, 0), (1992, 0), (1993, 0), (1994, 0), (1995, 0), (1996, 0), (1997, 0), (1998, 0), (1999, 0), (2000, 0), (2001, 0), (2002, 0), (2003, 0), (2004, 0), (2005, 0), (2006, 0), (2007, 0), (2008, 0), (2009, 0), (2010, 0), (2011, 0), (2012, 0), (2013, 0), (2014, 0)) ~~|

Historical investment subsidy[California]([(0,0)-(2014,10)], (1980, 0.9502), (1981, 0.9502), (1982, 0.9502), (1983, 0.9502), (1984, 0.9502), (1985, 0.9502), (1986, 0), (1987, 0), (1988, 0), (1989, 0), (1990, 0), (1991, 0), (1992, 0), (1993, 0), (1994, 0), (1995, 0), (1996, 0), (1997, 0), (1998, 0), (1999, 0), (2000, 0), (2001, 0), (2002, 0), (2003, 0), (2004, 0), (2005, 0), (2006, 0), (2007, 0), (2008, 0), (2009, 0), (2010, 0), (2011, 0), (2012, 0), (2013, 0), (2014, 0))

Units: dimensionless

Description: Lookup function for investment subsidy based on year.

Historical feed in tarriff rate[Denmark]([(0,0)-(2014,10)], (1980, 0), (1981, 0), (1982, 0), (1983, 0), (1984, 0), (1985, 0), (1986, 0), (1987, 0), (1988, 0), (1989, 0), (1990, 0), (1991, 0), (1992, 0), (1993, 0), (1994, 0), (1995, 0), (1996, 0), (1997, 0), (1998, 0), (1999, 0), (2000, 0), (2001, 0), (2002, 0), (2003, 0), (2004, 0), (2005, 0), (2006, 0), (2007, 0), (2008, 0), (2009, 0), (2010, 0), (2011, 0), (2012, 0), (2013, 0), (2014, 0)) ~~|

Historical feed in tarriff rate[Germany]([(0,0)-(2014,10)], (1980, 0), (1981, 0), (1982, 0), (1983, 0), (1984, 0), (1985, 0), (1986, 0), (1987, 0), (1988, 0), (1989, 0), (1990, 0), (1991, 0), (1992, 0), (1993, 0), (1994, 0.10956), (1995, 0.10974), (1996, 0.094971), (1997, 0.095032), (1998, 0.09375), (1999, 0.090271), (2000, 0.075012), (2001, 0.09436), (2002, 0.087891), (2003, 0.10254), (2004, 0.11987), (2005, 0.12512), (2006, 0.10516), (2007, 0.11438), (2008, 0.09845), (2009, 0.11639), (2010, 0.11865), (2011, 0.10706), (2012, 0.10095), (2013, 0.10181), (2014, 0.10474)) ~~|

Historical feed in tarriff rate[Spain]([(0,0)-(2014,10)], (1980, 0), (1981, 0), (1982, 0), (1983, 0), (1984, 0), (1985, 0), (1986, 0), (1987, 0), (1988, 0), (1989, 0), (1990, 0), (1991, 0), (1992, 0), (1993, 0), (1994, 0.094727), (1995, 0.094727), (1996, 0.094727), (1997, 0.094727), (1998, 0.094727), (1999, 0.094727), (2000, 0.094727), (2001, 0.085632), (2002, 0.079773), (2003, 0.093079), (2004, 0.10883), (2005,

0.11359), (2006, 0.095459), (2007, 0.10382), (2008, 0.089417), (2009, 0.10571), (2010, 0.10773), (2011, 0.097168), (2012, 0), (2013, 0), (2014, 0)) ~~|

Historical feed in tarriff rate[California]([(0,0)-(2014,10)], (1980, 0), (1981, 0), (1982, 0), (1983, 0), (1984, 0), (1985, 0), (1986, 0), (1987, 0), (1988, 0), (1989, 0), (1990, 0), (1991, 0), (1992, 0), (1993, 0), (1994, 0), (1995, 0), (1996, 0), (1997, 0), (1998, 0), (1999, 0), (2000, 0), (2001, 0), (2002, 0), (2003, 0), (2004, 0), (2005, 0), (2006, 0), (2007, 0), (2008, 0.089233), (2009, 0.089233), (2010, 0.089233), (2011, 0.089233), (2012, 0.089233), (2013, 0.089233), (2014, 0.089233))

Units: $/ (kW * hrs)

Description: Lookup function for feed-in-tariff based on year.

Historical rec price[Denmark]([(0,0)-(2014,10)], (1980, 0), (1981, 0), (1982, 0), (1983, 0), (1984, 0), (1985, 0), (1986, 0), (1987, 0), (1988, 0), (1989, 0), (1990, 0), (1991, 0), (1992, 0), (1993, 0), (1994, 0), (1995, 0), (1996, 0), (1997, 0), (1998, 0), (1999, 0), (2000, 0), (2001, 0), (2002, 0), (2003, 0), (2004, 0), (2005, 0), (2006, 0), (2007, 0), (2008, 0), (2009, 0), (2010, 0), (2011, 0), (2012, 0), (2013, 0), (2014, 0)) ~~|

Historical rec price[Germany]([(0,0)-(2014,10)], (1980, 0), (1981, 0), (1982, 0), (1983, 0), (1984, 0), (1985, 0), (1986, 0), (1987, 0), (1988, 0), (1989, 0), (1990, 0), (1991, 0), (1992, 0), (1993, 0), (1994, 0), (1995, 0), (1996, 0), (1997, 0), (1998, 0), (1999, 0), (2000, 0), (2001, 0), (2002, 0), (2003, 0), (2004, 0), (2005, 0), (2006, 0), (2007, 0), (2008, 0), (2009, 0), (2010, 0), (2011, 0), (2012, 0), (2013, 0), (2014, 0)) ~~|

Historical rec price[Spain]([(0,0)-(2014,10)], (1980, 0), (1981, 0), (1982, 0), (1983, 0), (1984, 0), (1985, 0), (1986, 0), (1987, 0), (1988, 0), (1989, 0), (1990, 0), (1991, 0), (1992, 0), (1993, 0), (1994, 0), (1995, 0), (1996, 0), (1997, 0), (1998, 0), (1999, 0), (2000, 0), (2001, 0), (2002, 0), (2003, 0), (2004, 0), (2005, 0), (2006, 0), (2007, 0), (2008, 0), (2009, 0), (2010, 0), (2011, 0), (2012, 0), (2013, 0), (2014, 0)) ~~|

Historical rec price[California]([(0,0)-(2014,10)], (1980, 0), (1981, 0), (1982, 0), (1983, 0), (1984, 0), (1985, 0), (1986, 0), (1987, 0), (1988, 0), (1989, 0), (1990, 0), (1991, 0), (1992, 0), (1993, 0), (1994, 0), (1995, 0), (1996, 0), (1997, 0), (1998, 0), (1999, 0), (2000, 0), (2001, 0), (2002, 0), (2003, 0.029999), (2004, 0.029999), (2005, 0.029999), (2006, 0.029999), (2007, 0.029999), (2008, 0.029999), (2009, 0.029999), (2010, 0.010002), (2011, 0.010002), (2012, 0.010002), (2013, 0.010002), (2014, 0.010002))

Units: $/ (kW * hrs)

Description: Lookup function for REC price based on year.

Historical Turbine Cost([(0,0)-(2015,10)], (1980, 2000), (1981, 2036.54), (1982, 1936.42), (1983, 1905.41), (1984, 1874.92), (1985, 1844.93), (1986, 2041.8), (1987, 1725.8), (1988, 1757.89), (1989, 1729.83), (1990, 1702.23), (1991, 1675.08), (1992, 1648.38), (1993, 1622.12), (1994, 1596.29), (1995, 1570.88), (1996, 1545.89), (1997, 1502.54), (1998, 1178.1), (1999, 1473.35), (2000, 760.52), (2001,

837.318), (2002, 791.782), (2003, 1045.46), (2004, 1075.15), (2005, 1287.48), (2006, 1322.57), (2007, 1476.2), (2008, 1648.43), (2009, 1505.34), (2010, 1515.28), (2011, 1380.17), (2012, 1214.74), (2013, 1220.57), (2014, 1086.04))

Units: $/ kW

Description: Historical turbine costs per kW.

Historical BOS Costs([(0,0)-(2015,10)], (1980, 1000), (1981, 984.469), (1982, 969.187), (1983, 954.15), (1984, 939.354), (1985, 924.794), (1986, 948.965), (1987, 481.268), (1988, 882.497), (1989, 868.846), (1990, 855.412), (1991, 842.193), (1992, 829.185), (1993, 816.384), (1994, 803.787), (1995, 791.391), (1996, 779.192), (1997, 1024.56), (1998, 595.712), (1999, 743.745), (2000, 1006.34), (2001, 589.635), (2002, 708.241), (2003, 484.072), (2004, 616.975), (2005, 275.539), (2006, 446.777), (2007, 362.334), (2008, 426.804), (2009, 864.04), (2010, 1017.56), (2011, 1306.62), (2012, 1265.04), (2013, 1098.14), (2014, 746.772))

Units: $/ kW

Description: Historical BOS costs per kW.

Historical OPEX Costs([(0,0)-(2015,10)], (1980, 50), (1981, 47.4182), (1982, 44.9709), (1983, 42.6511), (1984, 39.3374), (1985, 52.7259), (1986, 36.391), (1987, 41.1144), (1988, 32.7412), (1989, 22.2892), (1990, 20.9639), (1991, 27.9468), (1992, 26.5115), (1993, 25.1506), (1994, 23.4488), (1995, 22.6366), (1996, 21.4763), (1997, 20.376), (1998, 20.4819), (1999, 26.5301), (2000, 11.3856), (2001, 10.327), (2002, 9.47504), (2003, 22.5), (2004, 15.5422), (2005, 10.4819), (2006, 12.1988), (2007, 23.9639), (2008, 10.753), (2009, 11.5662), (2010, 10.4819), (2011, 8.26808), (2012, 11.7018), (2013, 8.81493), (2014, 8.36702))

Units: $/ (kW * hrs)

Description: Historical OPEX costs per kWh.

Historical Marginal LCOE([(0,0)-(2015,10)], (1980, 0.271425), (1981, 0.251373), (1982, 0.232811), (1983, 0.184569), (1984, 0.199722), (1985, 0.141141), (1986, 0.171362), (1987, 0.119427), (1988, 0.147051), (1989, 0.10857), (1990, 0.10857), (1991, 0.10857), (1992, 0.108338), (1993, 0.097713), (1994, 0.0930112), (1995, 0.0814275), (1996, 0.0814275), (1997, 0.075999), (1998, 0.0685875), (1999, 0.0597135), (2000, 0.0589114), (2001, 0.054285), (2002, 0.0506081), (2003, 0.054285), (2004, 0.056),

(2005, 0.056), (2006, 0.07), (2007, 0.071), (2008, 0.084), (2009, 0.088), (2010, 0.08), (2011, 0.064), (2012, 0.057), (2013, 0.046), (2014, 0.041))

Units: $/ (kW * hrs)

Description: Historical LCOE in $/kWh.

Historical Marginal Rotor Diameter([(0,0)-(2015,10)], (1980, 10), (1981, 10), (1982, 10), (1983, 10), (1984, 18), (1985, 18), (1986, 18), (1987, 18), (1988, 24.2424), (1989, 25.4512), (1990, 27.5012), (1991, 26.6654), (1992, 28.2681), (1993, 29.6379), (1994, 33.8287), (1995, 38.169), (1996, 38.834), (1997, 40.6219), (1998, 43.4687), (1999, 49.0227), (2000, 48.3811), (2001, 54.8742), (2002, 57.192), (2003, 65.1393), (2004, 64.4224), (2005, 70.2061), (2006, 69.6005), (2007, 71.6756), (2008, 74.6509), (2009, 77.3713), (2010, 82.7534), (2011, 85.0482), (2012, 91.5674), (2013, 91.265), (2014, 97.3853))

Units: meters

Description: Historical rotor diameter in m.

Historical Marginal Rated Power([(0,0)-(2015,10)], (1980, 22), (1981, 22), (1982, 22), (1983, 22), (1984, 55), (1985, 55), (1986, 55), (1987, 75), (1988, 153.792), (1989, 50.4832), (1990, 97.4881), (1991, 149.661), (1992, 155.164), (1993, 197.7), (1994, 173.555), (1995, 179.557), (1996, 355.797), (1997, 372.782), (1998, 463.271), (1999, 568.656), (2000, 528.066), (2001, 652.357), (2002, 774.455), (2003, 1026.68), (2004, 1067.68), (2005, 1208.29), (2006, 1147.47), (2007, 1323.28), (2008, 1474.49), (2009, 1512.12), (2010, 1631.75), (2011, 1646.71), (2012, 1795.61), (2013, 1665.22), (2014, 1598.75))

Units: kW

Description: Historical rated power in kW.

Historical Marginal Hub Height([(0,0)-(2015,10)], (1980, 15), (1981, 15), (1982, 15), (1983, 15), (1984, 18), (1985, 18), (1986, 18), (1987, 18), (1988, 29.1709), (1989, 30.9923), (1990, 31.125), (1991, 31.3598), (1992, 31.9889), (1993, 29.8407), (1994, 29.7731), (1995, 42.313), (1996, 42.6721), (1997, 44.2672), (1998, 47.8523), (1999, 55.2681), (2000, 55.5047), (2001, 60.0999), (2002, 65.024), (2003, 70.6356), (2004, 73.7505), (2005, 77.5187), (2006, 77.9399), (2007, 78.9457), (2008, 78.7889), (2009, 80.7106), (2010, 81.3083), (2011, 83.8566), (2012, 86.0163), (2013, 92.8536), (2014, 101.28))

Units: meters

Description: Historical hub height in m.

Historical Turbine Construction Rate[Denmark]([(0,0)-(2015,10)], (1980, 0), (1981, 0), (1982, 0), (1983, 0), (1984, 0), (1985, 0), (1986, 1), (1987, 12), (1988, 120), (1989, 140), (1990, 628), (1991, 194), (1992, 98), (1993, 89), (1994, 114), (1995, 136), (1996, 380), (1997, 516), (1998, 459), (1999, 409), (2000, 724), (2001, 122), (2002, 367), (2003, 118), (2004, 6), (2005, 21), (2006, 3), (2007, 3), (2008, 32), (2009, 157), (2010, 141), (2011, 82), (2012, 225), (2013, 113), (2014, 7), (2015, 6), (2016, 0)) ~~|

Historical Turbine Construction Rate[Germany]([(0,0)-(2015,10)], (1980, 0), (1981, 0), (1982, 0), (1983, 1), (1984, 0), (1985, 0), (1986, 0), (1987, 1), (1988, 0), (1989, 0), (1990, 23), (1991, 47), (1992, 75), (1993, 172), (1994, 323), (1995, 570), (1996, 498), (1997, 523), (1998, 889), (1999, 1210), (2000, 2000), (2001, 2726), (2002, 2220), (2003, 1903), (2004, 1288), (2005, 1119), (2006, 1417), (2007, 959), (2008, 569), (2009, 1385), (2010, 519), (2011, 528), (2012, 440), (2013, 493), (2014, 664), (2015, 736), (2016, 0)) ~~|

Historical Turbine Construction Rate[Spain]([(0,0)-(2015,10)], (1980, 0), (1981, 0), (1982, 0), (1983, 0), (1984, 0), (1985, 0), (1986, 0), (1987, 0), (1988, 0), (1989, 0), (1990, 0), (1991, 0), (1992, 0), (1993, 0), (1994, 0), (1995, 27), (1996, 1), (1997, 130), (1998, 211), (1999, 96), (2000, 220), (2001, 152), (2002, 349), (2003, 193), (2004, 447), (2005, 325), (2006, 189), (2007, 357), (2008, 252), (2009, 250), (2010, 49), (2011, 84), (2012, 108), (2013, 16), (2014, 2), (2015, 7), (2016, 0))~~|

Historical Turbine Construction Rate[California]([(0,0)-(2015,10)], (1980, 0), (1981, 0), (1982, 0), (1983, 144), (1984, 1343), (1985, 1074), (1986, 135), (1987, 648), (1988, 0), (1989, 336), (1990, 396), (1991, 0), (1992, 31), (1993, 322), (1994, 20), (1995, 6), (1996, 0), (1997, 56), (1998, 128), (1999, 339), (2000, 0), (2001, 112), (2002, 164), (2003, 161), (2004, 102), (2005, 34), (2006, 357), (2007, 21), (2008, 136), (2009, 94), (2010, 221), (2011, 236), (2012, 596), (2013, 176), (2014, 4), (2015, 96), (2016, 0))

Units: turbines/Year

Description: Lookup function for turbine construction rate in a given state by year.

Historical Turbine Installed Base[Denmark]([(0,0)-(2015,10)], (1980, 0), (1981, 0), (1982, 0), (1983, 0), (1984, 0), (1985, 0), (1986, 1), (1987, 13), (1988, 133), (1989, 273), (1990, 901), (1991, 1095), (1992, 1193), (1993, 1282), (1994, 1396), (1995, 1532), (1996, 1912), (1997, 2428), (1998, 2887), (1999, 3296), (2000, 4020), (2001, 4142), (2002, 4509), (2003, 4627), (2004, 4633), (2005, 4654), (2006, 4657), (2007, 4660), (2008, 4692), (2009, 4849), (2010, 4990), (2011, 5072), (2012, 5297), (2013, 5410), (2014, 5417), (2015, 5423), (2016, 5423)) ~~|

Historical Turbine Installed Base[Germany]([(0,0)-(2015,10)], (1980, 0), (1981, 0), (1982, 0), (1983, 1), (1984, 1), (1985, 1), (1986, 1), (1987, 2), (1988, 2), (1989, 2), (1990, 25), (1991, 72), (1992, 147), (1993, 319), (1994, 642), (1995, 1212), (1996, 1710), (1997, 2233), (1998, 3122), (1999, 4332), (2000, 6332), (2001, 9058), (2002, 11278), (2003, 13181), (2004, 14469), (2005, 15588), (2006, 17005), (2007, 17964), (2008, 18533), (2009, 19918), (2010, 20437), (2011, 20965), (2012, 21405), (2013, 21898), (2014, 22562), (2015, 23298), (2016, 23298)) ~~|

Historical Turbine Installed Base[Spain]([(0,0)-(2015,10)], (1980, 0), (1981, 0), (1982, 0), (1983, 0), (1984, 0), (1985, 0), (1986, 0), (1987, 0), (1988, 0), (1989, 0), (1990, 0), (1991, 0), (1992, 0), (1993, 0), (1994, 0), (1995, 27), (1996, 28), (1997, 158), (1998, 369), (1999, 465), (2000, 685), (2001, 837), (2002, 1186), (2003, 1379), (2004, 1826), (2005, 2151), (2006, 2340), (2007, 2697), (2008, 2949), (2009, 3199), (2010, 3248), (2011, 3332), (2012, 3440), (2013, 3456), (2014, 3458), (2015, 3465), (2016, 3465)) ~~|

Historical Turbine Installed Base[California]([(0,0)-(2015,10)], (1980, 0), (1981, 0), (1982, 0), (1983, 144), (1984, 1487), (1985, 2561), (1986, 2696), (1987, 3344), (1988, 3344), (1989, 3680), (1990, 4076), (1991, 4076), (1992, 4107), (1993, 4429), (1994, 4449), (1995, 4455), (1996, 4455), (1997, 4511), (1998, 4639), (1999, 4978), (2000, 4978), (2001, 5090), (2002, 5254), (2003, 5415), (2004, 5517), (2005, 5551), (2006, 5908), (2007, 5929), (2008, 6065), (2009, 6159), (2010, 6380), (2011, 6616), (2012, 7212), (2013, 7388), (2014, 7392), (2015, 7488), (2016, 7488))

Units: turbines

Description: Lookup function for total turbines installed in a given state by year.

Historical Capacity Installed Base[Denmark]([(0,0)-(2015,10)], (1980, 0.005), (1981, 0.007), (1982, 0.012), (1983, 0.02), (1984, 0.027), (1985, 0.05), (1986, 0.082), (1987, 0.115), (1988, 0.197), (1989, 0.262), (1990, 0.343), (1991, 0.413), (1992, 0.458), (1993, 0.491), (1994, 0.534725), (1995, 0.618725), (1996, 0.844725), (1997, 1.1369), (1998, 1.45194), (1999, 1.76794), (2000, 2.40464), (2001, 2.51267), (2002, 3.02271), (2003, 3.26666), (2004, 3.27466), (2005, 3.30426), (2006, 3.31026), (2007, 3.31251), (2008, 3.37286), (2009, 3.73676), (2010, 4.08576), (2011, 4.29861), (2012, 5.05071), (2013, 5.69571), (2014, 5.73371), (2015, 5.95301), (2016, 5.95301)) ~~|

Historical Capacity Installed Base[Germany]([(0,0)-(2015,10)], (1980, 0), (1981, 0), (1982, 0), (1983, 0.0003), (1984, 0.0003), (1985, 0.0003), (1986, 0.0003), (1987, 0.0053), (1988, 0.0153), (1989, 0.0273), (1990, 0.0483), (1991, 0.1103), (1992, 0.1833), (1993, 0.3343), (1994, 0.6433), (1995, 1.1373), (1996, 1.5643), (1997, 1.9663), (1998, 2.75519), (1999, 4.22119), (2000, 6.17819), (2001, 9.62409), (2002, 12.9557), (2003, 16.0153), (2004, 18.2312), (2005, 20.2203), (2006, 22.9275), (2007, 24.7881), (2008, 26.4441), (2009, 29.3266), (2010, 30.7636), (2011, 32.6096), (2012, 34.8816), (2013, 37.7996), (2014, 42.7146), (2015, 48.4966), (2016, 48.4966)) ~~|

Historical Capacity Installed Base[Spain]([(0,0)-(2015,10)], (1980, 0), (1981, 0), (1982, 0), (1983, 0), (1984, 0), (1985, 0), (1986, 0), (1987, 0), (1988, 0), (1989, 0), (1990, 0.002), (1991, 0.003), (1992, 0.033), (1993, 0.034), (1994, 0.041), (1995, 0.098), (1996, 0.227), (1997, 0.42), (1998, 0.848), (1999, 1.613), (2000, 2.206), (2001, 3.397), (2002, 4.891), (2003, 5.945), (2004, 8.317), (2005, 9.918), (2006, 11.722), (2007, 15.097), (2008, 16.74), (2009, 19.149), (2010, 20.676), (2011, 21.674), (2012, 22.796), (2013, 22.959), (2014, 22.987), (2015, 23.025), (2016, 23.025)) ~~|

Historical Capacity Installed Base[California]([(0,0)-(2015,10)], (1980, 0), (1981, 0), (1982, 0), (1983, 0.00576), (1984, 0.14006), (1985, 0.25008), (1986, 0.263118), (1987, 0.33957), (1988, 0.33957), (1989,

0.4381), (1990, 0.54307), (1991, 0.54307), (1992, 0.55004), (1993, 0.62699), (1994, 0.63699), (1995, 0.63999), (1996, 0.63999), (1997, 0.65784), (1998, 0.71024), (1999, 0.95571), (2000, 0.95571), (2001, 1.02281), (2002, 1.16278), (2003, 1.37836), (2004, 1.46889), (2005, 1.5308), (2006, 1.77972), (2007, 1.84272), (2008, 2.08172), (2009, 2.34297), (2010, 2.79762), (2011, 3.46202), (2012, 5.08702), (2013, 5.37852), (2014, 5.48492), (2015, 5.67592), (2016, 5.67592))

Units: GW

Description: Lookup function for total installed capacity in a given state by year.

Historical Capacity Construction Rate[Denmark]([(0,0)-(2015,10)], (1980, 0.005), (1981, 0.002), (1982, 0.005), (1983, 0.008), (1984, 0.007), (1985, 0.023), (1986, 0.032), (1987, 0.033), (1988, 0.082), (1989, 0.065), (1990, 0.081), (1991, 0.07), (1992, 0.045), (1993, 0.033), (1994, 0.043725), (1995, 0.084), (1996, 0.226), (1997, 0.292175), (1998, 0.315043), (1999, 0.316), (2000, 0.636699), (2001, 0.10803), (2002, 0.51004), (2003, 0.24395), (2004, 0.008), (2005, 0.0296), (2006, 0.006), (2007, 0.00225), (2008, 0.06035), (2009, 0.3639), (2010, 0.349), (2011, 0.21285), (2012, 0.7521), (2013, 0.645), (2014, 0.038), (2015, 0.2193), (2016, 0)) ~~|

Historical Capacity Construction Rate[Germany]([(0,0)-(2015,10)], (1980, 0), (1981, 0), (1982, 0), (1983, 0.0003), (1984, 0), (1985, 0), (1986, 0), (1987, 0.005), (1988, 0.01), (1989, 0.012), (1990, 0.021), (1991, 0.062), (1992, 0.073), (1993, 0.151), (1994, 0.309), (1995, 0.494), (1996, 0.427), (1997, 0.402), (1998, 0.78889), (1999, 1.466), (2000, 1.957), (2001, 3.44589), (2002, 3.3316), (2003, 3.05961), (2004, 2.21587), (2005, 1.98917), (2006, 2.70711), (2007, 1.86065), (2008, 1.656), (2009, 2.88245), (2010, 1.437), (2011, 1.846), (2012, 2.272), (2013, 2.918), (2014, 4.915), (2015, 5.782), (2016, 0)) ~~|

Historical Capacity Construction Rate[Spain]([(0,0)-(2015,10)], (1980, 0), (1981, 0), (1982, 0), (1983, 0), (1984, 0), (1985, 0), (1986, 0), (1987, 0), (1988, 0), (1989, 0), (1990, 0.002), (1991, 0.001), (1992, 0.03), (1993, 0.001), (1994, 0.007), (1995, 0.057), (1996, 0.129), (1997, 0.193), (1998, 0.428), (1999, 0.765), (2000, 0.593), (2001, 1.191), (2002, 1.494), (2003, 1.054), (2004, 2.372), (2005, 1.601), (2006, 1.804), (2007, 3.375), (2008, 1.643), (2009, 2.409), (2010, 1.527), (2011, 0.998), (2012, 1.122), (2013, 0.163), (2014, 0.028), (2015, 0.038), (2016, 0)) ~~|

Historical Capacity Construction Rate[California]([(0,0)-(2015,10)], (1980, 0), (1981, 0), (1982, 0), (1983, 0.00576), (1984, 0.1343), (1985, 0.11002), (1986, 0.013038), (1987, 0.076452), (1988, 0), (1989, 0.09853), (1990, 0.10497), (1991, 0), (1992, 0.00697), (1993, 0.07695), (1994, 0.01), (1995, 0.003), (1996, 0), (1997, 0.01785), (1998, 0.0524), (1999, 0.24547), (2000, 0), (2001, 0.0671), (2002, 0.139974), (2003, 0.21558), (2004, 0.09053), (2005, 0.06191), (2006, 0.24892), (2007, 0.063), (2008, 0.239), (2009, 0.261246), (2010, 0.454654), (2011, 0.664393), (2012, 1.625), (2013, 0.2915), (2014, 0.1064), (2015, 0.191), (2016, 0))

Units: GW/Year

Description: Lookup function for capacity construction rate in a given state by year.

Historical rest of world turbine construction start rate([(0,0)-(2015,10)], (1980, 0), (1981, 0), (1982, 0), (1983, 10), (1984, 0), (1985, 1), (1986, 1), (1987, 0), (1988, 0), (1989, 31), (1990, 4), (1991, 10), (1992, 223), (1993, 243), (1994, 304), (1995, 231), (1996, 640), (1997, 551), (1998, 752), (1999, 1591), (2000, 1407), (2001, 2886), (2002, 1856), (2003, 2537), (2004, 2109), (2005, 4263), (2006, 6283), (2007, 8541), (2008, 10431), (2009, 13212), (2010, 9631), (2011, 7653), (2012, 9517), (2013, 5333), (2014, 8686))

Units: turbines/Year

Description: Lookup function for turbine construction rate for the rest of the world not included in the State group set by year.

Python Script for data pre-processing

```
#!/usr/bin/env python
# encoding: utf-8

"""
readingexceldata2016.py
Created by Katherine Dykes 2010.
Copyright (c) Katherine Dykes. All rights reserved.

This script analyzes a workbook of historical data for the wind energy and parses it
to develop code for the system dynamics model VENSIM.
The current script is calibrated to output data for 1980 to 2014 inclusive.  If the
underlying workbook and data period are updated, the script must be updated in
several places.

To use the script, first set the group of interest (variable = group).  A custom
group can be created in the "States" worksheet of the workbook by adding a new
column.

Specify which variable set to run (if you have a large number of groups, then running
full script can take a long time.  You may want to run the full script once but then
only run parts of it as necessary if specific areas of workbook have been updated.

Also specify if you would like to run a forward looking policy scenario:
  - policy 1 will create a set-up where policy is held constant to 2030 after 2014.
  - policy 2 will create a set-up where policy support is completely removed after
2014.
"""

import xlrd as EXCELREADER
import numpy as NP
import operator as OP
import unicodedata as UC

########
# Step 0 - select which groups, variables and policies for VENSIM that you are
updating; get workbook data
```

```
group = 5 # needs to be specified when run to get appropriate output; this is the
group according to the excel sheet "States"

doland = True # land area will be run if TRUE
dowind = True # wind resource will be run if TRUE
doelec = True # electricity prices will be run if TRUE
dopolicy = True # policy incentives will be run if TRUE
docosttech = True # historical cost and tech trends will be run if TRUE
dohistoric = True # historic capacity and turbine will be run if TRUE

policy0 = False # set to true if generation the set-up for policy 0 which disables
policy supports prior to 2000
policy1 = False # set to true if generating the set-up for policy 1 which keeps
policy constant from 2015 to 2030
policy2 = False # set to true of generating the set-up for policy 2 which keeps
policy at 0 after 2014

# open data workbook and assign all the necessary sheets to variables
book = EXCELREADER.open_workbook("wind data - compiled 2015.xlsx")
statesheet = book.sheet_by_name('States')
windsheet = book.sheet_by_name('WindData')
projectsheet = book.sheet_by_name('Windfarms')
elecsheet = book.sheet_by_name('Electricity')
costsheet = book.sheet_by_name('Costs')
techsheet = book.sheet_by_name('Technology')
policyPIsheet = book.sheet_by_name('PolicyPI')
policyIIsheet = book.sheet_by_name('PolicyII')
policyRECsheet = book.sheet_by_name('PolicyREC')
policyFITsheet = book.sheet_by_name('PolicyFIT')
totalsheet = book.sheet_by_name('Totals')

########
# Step 1 - Get groups for subsequent data analysis; Procedure for identifying which
provinces and countries are joined with which groups
group_col = 8 + group

# find how many items are indexed - those with cell value 0 are excluded
count = 0
for x in range(1, statesheet.nrows):
  if statesheet.cell_value(x, group_col) != 0:
    count = count + 1
instring = OP.concat(OP.concat(str(count),'i8, '),OP.concat(str(count),'a25'))
group_array = NP.zeros(1,dtype=instring)

# extract groups with entry id and associated group id
count = 0
num_groups = 0
num_states = 0
for x in range(1, statesheet.nrows):
  if statesheet.cell_value(x,group_col) != 0:
    num_states = num_states + 1
    group_array[0][0][count] = statesheet.cell_value(x,0)
    if statesheet.cell_value(x,group_col) in group_array[0][1][:]:
      group_array[0][1][count] = statesheet.cell_value(x,group_col)
    else:
```

```
      group_array[0][1][count] = statesheet.cell_value(x,group_col)
      num_groups = num_groups + 1
    count = count + 1

instring = OP.concat(str(num_groups),'a25')
group_list = NP.zeros(1,dtype=instring)
count = 0
year_col = 5
for x in range(1, statesheet.nrows):
  if statesheet.cell_value(x,group_col) in group_list[0,:]:
    count = count
  elif statesheet.cell_value(x,group_col) != 0:
    if count < num_groups:
      group_list[0,count] = statesheet.cell_value(x,group_col)
      count = count + 1

# now print the group list to create the subscript functions for VENSIM
count = 0
f = open('vensiminputtxt.txt','w')
f.write('States:\n')
for x in range(0,num_groups):
  f.write(group_list[0,count])
  if count < (num_groups - 1):
    f.write(', ')
  count = count + 1
  if count < num_groups - 1:
    if OP.mod(count,10) == 0:
      f.write('\\\n')
f.write('\n\t~\n\t~\t States in the analysis.\t|')

count = 0
f.write('\n\n')
f.write('Statescopy:\n')
for x in range(0,num_groups):
  f.write(group_list[0,count])
  if count < (num_groups - 1):
    f.write(', ')
  count = count + 1
  if count < num_groups - 1:
    if OP.mod(count,10) == 0:
      f.write('\\\n')
f.write('\n\t~\n\t~\t Copy of States in the analysis for simulation.\t|')

#########
# Step 2: Collect all the necessary variables for VENSIM for each identified group

# FUNCTION - Find total potential land for each group
# Procedure for aggregating total land area for each group as defined in Get groups
function
if doland:
  area_col = 6
  instring = OP.concat(OP.concat(str(num_groups),'a25,
'),OP.concat(str(num_groups),'f4'))
  area_array = NP.zeros(1,dtype=instring)
  for x in range(0,num_groups):
```

```
    area_array[0][0][x] = group_list[0,x]
    for y in range(1, statesheet.nrows):
      if statesheet.cell_value(y,group_col) == group_list[0,x]:
        area_array[0][1][x] = area_array[0][1][x] +
statesheet.cell_value(y,area_col)*1000*1000

  # now print the area to the create the subscript functions for VENSIM
  count = 0
  f.write('\n\n')
  for x in range(0,num_groups):
    f.write('total potential land[')
    f.write(area_array[0][0][x])
    f.write(']= ')
    f.write(str(area_array[0][1][x]))
    if count < (num_groups - 1):
      f.write('~~|\n')
    count = count + 1
  f.write('\n\t~\t meters * meters\n')
  f.write('\t~\t Total available land area in a given state.\t|\n')


# FUNCTION - Find wind data by state
# Procedure of finding the wind resource polynomial fit coefficients for a region
if dowind:
  wind_col = 5
  state_col = 3
  prov_col = 4
  instring = OP.concat(OP.concat(str(num_groups),'a25,
'),OP.concat(str(num_groups),'f4'))
  max_array = NP.zeros(1,dtype=instring)
  linear_array = NP.zeros(1,dtype=instring)
  quad_array = NP.zeros(1,dtype=instring)
  for x in range(0,num_groups):
    max_array[0][0][x] = group_list[0,x]
    linear_array[0][0][x] = group_list[0,x]
    quad_array[0][0][x] = group_list[0,x]
    windlist = []
    landlist = []
    count = 1.0
    for w in range(0,num_states):
      for z in range(1,windsheet.nrows):
        if windsheet.cell_value(z,state_col) == group_array[0][0][w]:
          if group_array[0][1][w] == group_list[0,x]:
            windlist.append(windsheet.cell_value(z,wind_col))
            landlist.append(count)
            count = count + 1
        elif windsheet.cell_value(z,prov_col) == group_array[0][0][w]:
          if group_array[0][1][w] == group_list[0,x]:
            windlist.append(windsheet.cell_value(z,wind_col))
            landlist.append(count)
            count = count + 1
    if count > 3:
      windreverse = sorted(windlist, reverse = True)
      windsorted = NP.array(windreverse)
      landordered = NP.array(landlist)
```

```
        landsorted = landordered / count
        coeff = NP.polyfit(landsorted, windsorted, 2)
        max_array[0][1][x] = windsorted[0]
        linear_array[0][1][x] = coeff[1]
        quad_array[0][1][x] = coeff[0]

    # now print the wind data to the create the subscript functions for VENSIM
    # # #
    count = 0
    f.write('\n\n')
    for x in range(0,num_groups):
      f.write('max wind speed[')
      f.write(max_array[0][0][x])
      f.write(']= ')
      f.write(str(max_array[0][1][x]))
      if count < (num_groups - 1):
        f.write('~~|\n')
      count = count + 1
    f.write('\n\t~\t')
    f.write('meters / second\n')
    f.write('\t~\t Maximum wind speed in a given region.\t|\n')

    # # #
    count = 0
    f.write('\n\n')
    for x in range(0,num_groups):
      f.write('linear term for wind speed by land use[')
      f.write(linear_array[0][0][x])
      f.write(']= ')
      f.write(str(linear_array[0][1][x]))
      if count < (num_groups - 1):
        f.write('~~|\n')
      count = count + 1
    f.write('\n\t~\t')
    f.write('meters / second\n')
    f.write('\t~\t Linear coefficient for polynomial fit of wind speed resource by land
use.\t|\n')

    # # #
    count = 0
    f.write('\n\n')
    for x in range(0,num_groups):
      f.write('quadratic term for wind speed by land use[')
      f.write(quad_array[0][0][x])
      f.write(']= ')
      f.write(str(quad_array[0][1][x]))
      if count < (num_groups - 1):
        f.write('~~|\n')
      count = count + 1
    f.write('\n\t~\t')
    f.write('meters / second\n')
    f.write('\t~\t Quadratic coefficient for polynomial fit of wind speed resource by
land use.\t|\n')
```

```
# FUNCTION - Find electricity prices by year for the group
# Procedure of finding the long term electricity price trends for a group
if doelec:
  col_start = 6
  col_end = 41
  state_col = 4
  prov_col = 5
  instring = OP.concat(OP.concat(str(num_groups),'a25,
('),OP.concat(OP.concat(str(col_end-
col_start),','),OP.concat(str(num_groups),')f2')))
  electricity_array = NP.zeros(1,dtype=instring)
  for x in range(0,num_groups):
    electricity_array[0][0][x] = group_list[0,x]
    for w in range(0,num_states):
      for z in range(1,elecsheet.nrows):
        stategroup = 0
        if elecsheet.cell_value(z,state_col) == group_array[0][0][w]:
          if group_array[0][1][w] == group_list[0,x]:
            stategroup = 1
            for y in range(0, col_end - col_start):
              if elecsheet.cell_value(z,col_start+y) != 0:
                electricity_array[0][1][y][x] = elecsheet.cell_value(z,col_start + y)
        if stategroup == 0:
          if elecsheet.cell_value(z,prov_col) == group_array[0][0][w]:
            if group_array[0][1][w] == group_list[0,x]:
              for y in range(0, col_end - col_start):
                if elecsheet.cell_value(z,col_start+y) != 0:
                  electricity_array[0][1][y][x] = elecsheet.cell_value(z,col_start +
y)

  # now print the electricity prices to the create the subscript functions for VENSIM
  count = 0
  startyr = 1980
  f.write('\n\n')
  for x in range(0,num_groups):
    trigger = 0
    f.write('Historical Electricity Prices[')
    f.write(electricity_array[0][0][x])
    f.write(']([(0,0)-(2014,10)]')
    for y in range(0, col_end - col_start):
      f.write(', (')
      f.write(str(startyr + y))
      f.write(', ')
      f.write(str(electricity_array[0][1][y][x]))
      f.write(')')
      trigger = trigger + 1
      if OP.mod(trigger,10) == 0:
        f.write('\\\n')
    f.write(') ')
    if count < (num_groups - 1):
      f.write('~~|\n')
    count = count + 1
  f.write('\n\t~\t$ / (kW * hrs)\n')
  f.write('\t~\tLookup function for electricity price based on year.\n\t|\n')
```

```
# FUNCTION - Find policy incentives by year for the group
# Procedure of finding the long term policy trends for a group
if dopolicy:

  # Production Incentive policies
  col_start = 6
  col_end = 41
  state_col = 4
  prov_col = 5
  instring = OP.concat(OP.concat(str(num_groups),'a25,
('),OP.concat(OP.concat(str(col_end-
col_start),','),OP.concat(str(num_groups),')f2')))
  policy_array = NP.zeros(1,dtype=instring)
  for x in range(0,num_groups):
    policy_array[0][0][x] = group_list[0,x]
    for w in range(0,num_states):
      for z in range(1,policyPIsheet.nrows):
        stategroup = 0
        if policyPIsheet.cell_value(z,state_col) == group_array[0][0][w]:
          if group_array[0][1][w] == group_list[0,x]:
            stategroup = 1
            for y in range(0, col_end - col_start):
              if policyPIsheet.cell_value(z,col_start+y) != 0:
                policy_array[0][1][y][x] = policyPIsheet.cell_value(z,col_start + y)
        if stategroup == 0:
          if policyPIsheet.cell_value(z,prov_col) == group_array[0][0][w]:
            if group_array[0][1][w] == group_list[0,x]:
              for y in range(0, col_end - col_start):
                if policyPIsheet.cell_value(z,col_start+y) != 0:
                  policy_array[0][1][y][x] = policyPIsheet.cell_value(z,col_start +
y)

  # now print the production incentives to the create the subscript functions for
VENSIM
  count = 0
  startyr = 1980
  f.write('\n\n')
  for x in range(0,num_groups):
    trigger = 0
    f.write('Historical production incentive[')
    f.write(policy_array[0][0][x])
    f.write(']([(0,0)-(2014,10)]')
    for y in range(0, col_end - col_start):
      f.write(', (')
      f.write(str(startyr + y))
      f.write(', ')
      if policy0:
        if startyr + y < 1990:
          f.write(str(0))
        else:
          f.write(str(policy_array[0][1][y][x]))
      else:
        f.write(str(policy_array[0][1][y][x]))
      f.write(')')
```

```
      trigger = trigger + 1
      if OP.mod(trigger,10) == 0:
        f.write('\\\n')
    if policy1:
      f.write(', (2030, ')
      f.write(str(policy_array[0][1][y][x]))
      f.write(')')
    if policy2:
      f.write(', (2015, 0.0)')
    f.write(') ')
    if count < (num_groups - 1):
      f.write('~~|\n')
    count = count + 1
  f.write('\n\t~\t')
  f.write('$ / (kW * hrs)\n\t~\tLookup function for production incentive based on
year.\n\t|\n')


  # Investment Incentive policies
  col_start = 6
  col_end = 41
  state_col = 4
  prov_col = 5
  instring = OP.concat(OP.concat(str(num_groups),'a25,
('),OP.concat(OP.concat(str(col_end-
col_start),','),OP.concat(str(num_groups),')f2')))
  policy_array = NP.zeros(1,dtype=instring)
  for x in range(0,num_groups):
    policy_array[0][0][x] = group_list[0,x]
    for w in range(0,num_states):
      for z in range(1,policyIIsheet.nrows):
        stategroup = 0
        if policyIIsheet.cell_value(z,state_col) == group_array[0][0][w]:
          if group_array[0][1][w] == group_list[0,x]:
            stategroup = 1
            for y in range(0, col_end - col_start):
              if policyIIsheet.cell_value(z,col_start+y) != 0:
                policy_array[0][1][y][x] = policyIIsheet.cell_value(z,col_start + y)
        if stategroup == 0:
          if policyIIsheet.cell_value(z,prov_col) == group_array[0][0][w]:
            if group_array[0][1][w] == group_list[0,x]:
              for y in range(0, col_end - col_start):
                if policyIIsheet.cell_value(z,col_start+y) != 0:
                  policy_array[0][1][y][x] = policyIIsheet.cell_value(z,col_start +
y)

  # now print the investment subsidies to the create the subscript functions for
VENSIM
  count = 0
  startyr = 1980
  f.write('\n\n')
  for x in range(0,num_groups):
    trigger = 0
    f.write('Historical investment subsidy[')
    f.write(policy_array[0][0][x])
```

```
    f.write(']([(0,0)-(2014,10)]')
    for y in range(0, col_end - col_start):
      f.write(', (')
      f.write(str(startyr + y))
      f.write(', ')
      if policy0:
        if startyr + y < 1990:
          f.write(str(0))
        else:
          f.write(str(policy_array[0][1][y][x]))
      else:
        f.write(str(policy_array[0][1][y][x]))
      f.write(')')
      trigger = trigger + 1
      if OP.mod(trigger,10) == 0:
        f.write('\\\n')
    if policy1:
      f.write(', (2030, ')
      f.write(str(policy_array[0][1][y][x]))
      f.write(')')
    if policy2:
      f.write(', (2015, 0.0)')
    f.write(') ')
    if count < (num_groups - 1):
      f.write('~~|\n')
    count = count + 1
  f.write('\n\t~\t')
  f.write('dimensionless\n\t~\tLookup function for investment subsidy based on
year.\n\t|\n')


  # Feed in Tariff policies
  col_start = 6
  col_end = 41
  state_col = 4
  prov_col = 5
  instring = OP.concat(OP.concat(str(num_groups),'a25,
('),OP.concat(OP.concat(str(col_end-
col_start),','),OP.concat(str(num_groups),')f2')))
  policy_array = NP.zeros(1,dtype=instring)
  for x in range(0,num_groups):
    policy_array[0][0][x] = group_list[0,x]
    for w in range(0,num_states):
      for z in range(1,policyFITsheet.nrows):
        stategroup = 0
        if policyFITsheet.cell_value(z,state_col) == group_array[0][0][w]:
          if group_array[0][1][w] == group_list[0,x]:
            stategroup = 1
            for y in range(0, col_end - col_start):
              if policyFITsheet.cell_value(z,col_start+y) != 0:
                policy_array[0][1][y][x] = policyFITsheet.cell_value(z,col_start + y)
        if stategroup == 0:
          if policyFITsheet.cell_value(z,prov_col) == group_array[0][0][w]:
            if group_array[0][1][w] == group_list[0,x]:
              for y in range(0, col_end - col_start):
```

```
                if policyFITsheet.cell_value(z,col_start+y) != 0:
                  policy_array[0][1][y][x] = policyFITsheet.cell_value(z,col_start +
y)

  # now print the feed in tariff rates to the create the subscript functions for
VENSIM
  count = 0
  startyr = 1980
  f.write('\n\n')
  for x in range(0,num_groups):
    trigger = 0
    f.write('Historical feed in tarriff rate[')
    f.write(policy_array[0][0][x])
    f.write(']([(0,0)-(2014,10)]')
    for y in range(0, col_end - col_start):
      f.write(', (')
      f.write(str(startyr + y))
      f.write(', ')
      if policy0:
        if startyr + y < 1990:
          f.write(str(0))
        else:
          f.write(str(policy_array[0][1][y][x]))
      else:
        f.write(str(policy_array[0][1][y][x]))
      f.write(')')
      trigger = trigger + 1
      if OP.mod(trigger,10) == 0:
        f.write('\\\n')
    if policy1:
      f.write(', (2030, ')
      f.write(str(policy_array[0][1][y][x]))
      f.write(')')
    if policy2:
      f.write(', (2015, 0.0)')
    f.write(') ')
    if count < (num_groups - 1):
      f.write('~~|\n')
    count = count + 1
  f.write('\n\t~\t')
  f.write('$ / (kW * hrs)\n\t~\tLookup function for feed-in-tariff based on
year.\n\t|\n')


  # REC Incentive policies
  col_start = 6
  col_end = 41
  state_col = 4
  prov_col = 5
  instring = OP.concat(OP.concat(str(num_groups),'a25,
('),OP.concat(OP.concat(str(col_end-
col_start),','),OP.concat(str(num_groups),')f2')))
  policy_array = NP.zeros(1,dtype=instring)
  for x in range(0,num_groups):
    policy_array[0][0][x] = group_list[0,x]
```

```
    for w in range(0,num_states):
      for z in range(1,policyRECsheet.nrows):
        stategroup = 0
        if policyRECsheet.cell_value(z,state_col) == group_array[0][0][w]:
          if group_array[0][1][w] == group_list[0,x]:
            stategroup = 1
            for y in range(0, col_end - col_start):
              if policyRECsheet.cell_value(z,col_start+y) != 0:
                policy_array[0][1][y][x] = policyRECsheet.cell_value(z,col_start + y)
        if stategroup == 0:
          if policyRECsheet.cell_value(z,prov_col) == group_array[0][0][w]:
            if group_array[0][1][w] == group_list[0,x]:
              for y in range(0, col_end - col_start):
                if policyRECsheet.cell_value(z,col_start+y) != 0:
                  policy_array[0][1][y][x] = policyRECsheet.cell_value(z,col_start +
y)

  # now print the REC prices to the create the subscript functions for VENSIM
  count = 0
  startyr = 1980
  f.write('\n\n')
  for x in range(0,num_groups):
    trigger = 0
    f.write('Historical rec price[')
    f.write(policy_array[0][0][x])
    f.write('](([(0,0)-(2014,10)]')
    for y in range(0, col_end - col_start):
      f.write(', (')
      f.write(str(startyr + y))
      f.write(', ')
      if policy0:
        if startyr + y < 1990:
          f.write(str(0))
        else:
          f.write(str(policy_array[0][1][y][x]))
      else:
        f.write(str(policy_array[0][1][y][x]))
      f.write(')')
      trigger = trigger + 1
      if OP.mod(trigger,10) == 0:
        f.write('\\\n')
    if policy1:
      f.write(', (2030, ')
      f.write(str(policy_array[0][1][y][x]))
      f.write(')')
    if policy2:
      f.write(', (2015, 0.0)')
    f.write(') ')
    if count < (num_groups - 1):
      f.write('~~|\n')
    count = count + 1
  f.write('\n\t~\t')
  f.write('$ / (kW * hrs)\n\t~\tLookup function for REC price based on year.\n\t|\n')
```

```
# FUNCTION - Find cost and technology trends
# Procedure of finding the cost and technology trends
if docosttech:
  years = 35
  turbcost_array = NP.zeros(35)
  boscost_array = NP.zeros(35)
  omcost_array = NP.zeros(35)
  lcoecost_array = NP.zeros(35)
  for x in range(0, 35):
    turbcost_array[x] = costsheet.cell_value(x+1,1)
    boscost_array[x] = costsheet.cell_value(x+1,2)
    omcost_array[x] = costsheet.cell_value(x+1,3)
    lcoecost_array[x] = costsheet.cell_value(x+1,4)

  rdtech_array = NP.zeros(35)
  rptech_array = NP.zeros(35)
  hhtech_array = NP.zeros(35)
  for x in range(0, 35):
    rdtech_array[x] = techsheet.cell_value(x+1,1)
    rptech_array[x] = techsheet.cell_value(x+1,2)
    hhtech_array[x] = techsheet.cell_value(x+1,3)

  f.write('\n\n')
  f.write('Historical Turbine Cost([(0,0)-(2015,10)]')
  for x in range(0,35):
    f.write(', (')
    f.write(str(1980 + x))
    f.write(', ')
    f.write(str(turbcost_array[x]))
    f.write(')')
  f.write(')\n\t~\t $ / kW \n\t~\t')
  f.write('Historical turbine costs per kW.\n')
  f.write('\t|\n')

  f.write('\n\n')
  f.write('Historical BOS Costs([(0,0)-(2015,10)]')
  for x in range(0,35):
    f.write(', (')
    f.write(str(1980 + x))
    f.write(', ')
    f.write(str(boscost_array[x]))
    f.write(')')
  f.write(')\n\t~\t $ / kW \n\t~\t')
  f.write('Historical BOS costs per kW.\n')
  f.write('\t|\n')

  f.write('\n\n')
  f.write('Historical OPEX Costs([(0,0)-(2015,10)]')
  for x in range(0,35):
    f.write(', (')
    f.write(str(1980 + x))
    f.write(', ')
    f.write(str(omcost_array[x]))
    f.write(')')
  f.write(')\n\t~\t $ / (kW * hrs) \n\t~\t')
```

```
  f.write('Historical OPEX costs per kWh.\n')
  f.write('\t|\n')

  f.write('\n\n')
  f.write('Historical Marginal LCOE([(0,0)-(2015,10)]')
  for x in range(0,35):
    f.write(', (')
    f.write(str(1980 + x))
    f.write(', ')
    f.write(str(lcoecost_array[x]))
    f.write(')')
  f.write(')\n\t~\t $ / (kW * hrs) \n\t~\t')
  f.write('Historical LCOE in $/kWh.\n')
  f.write('\t|\n')

  f.write('\n\n')
  f.write('Historical Marginal Rotor Diameter([(0,0)-(2015,10)]')
  for x in range(0,35):
    f.write(', (')
    f.write(str(1980 + x))
    f.write(', ')
    f.write(str(rdtech_array[x]))
    f.write(')')
  f.write(')\n\t~\t meters\n\t~\t')
  f.write('Historical rotor diameter in m.\n')
  f.write('\t|\n')

  f.write('\n\n')
  f.write('Historical Marginal Rated Power([(0,0)-(2015,10)]')
  for x in range(0,35):
    f.write(', (')
    f.write(str(1980 + x))
    f.write(', ')
    f.write(str(rptech_array[x]))
    f.write(')')
  f.write(')\n\t~\t kW\n\t~\t')
  f.write('Historical rated power in kW.\n')
  f.write('\t|\n')

  f.write('\n\n')
  f.write('Historical Marginal Hub Height([(0,0)-(2015,10)]')
  for x in range(0,35):
    f.write(', (')
    f.write(str(1980 + x))
    f.write(', ')
    f.write(str(hhtech_array[x]))
    f.write(')')
  f.write(')\n\t~\t meters\n\t~\t')
  f.write('Historical hub height in m.\n')
  f.write('\t|\n')


# Function to aggregate historical variables for wind projects and technology
# Procedure to aggregate historical variables for wind projects and technology by
groups of interest.
```

```
if dohistoric:
  state_col = 3
  prov_col = 4
  year_col = 6
  turb_col = 7
  cap_col = 8
  power_col = 9
  diam_col = 10
  col_start = 4
  col_end = 41
  instring = OP.concat(OP.concat(str(num_groups),'a25,
('),OP.concat(OP.concat(str(col_end-
col_start),','),OP.concat(str(num_groups),')f8')))
  newprojects_array = NP.zeros(1,dtype=instring)
  newturbines_array = NP.zeros(1,dtype=instring)
  newcapacity_array = NP.zeros(1,dtype=instring)
  newpower_array = NP.zeros(1,dtype=instring)
  indexpower_array = NP.zeros(1,dtype=instring)
  newdiameter_array = NP.zeros(1,dtype=instring)
  indexdiam_array = NP.zeros(1,dtype=instring)
  for x in range(0,num_groups):
    newturbines_array[0][0][x] = group_list[0,x]
    newcapacity_array[0][0][x] = group_list[0,x]
    newpower_array[0][0][x] = group_list[0,x]
    newdiameter_array[0][0][x] = group_list[0,x]
    for w in range(0,num_states):
      for z in range(1,projectsheet.nrows):
        stategroup = 0
        if projectsheet.cell_value(z,state_col) == group_array[0][0][w]:
          if group_array[0][1][w] == group_list[0,x]:
            stategroup = 1
            if (projectsheet.cell_value(z,year_col) != '') &
(projectsheet.cell_value(z,year_col) <= 2015):
              y = projectsheet.cell_value(z,year_col) - 1980
              newprojects_array[0][1][y][x] = newprojects_array[0][1][y][x] + 1
              if projectsheet.cell_value(z,cap_col) > 0:
                  newcapacity_array[0][1][y][x] = newcapacity_array[0][1][y][x] +
projectsheet.cell_value(z,cap_col)
              if projectsheet.cell_value(z,turb_col) > 0:
                newturbines_array[0][1][y][x] = newturbines_array[0][1][y][x] +
projectsheet.cell_value(z,turb_col)
                if projectsheet.cell_value(z,power_col) > 0:
                  newpower_array[0][1][y][x] = newpower_array[0][1][y][x] +
projectsheet.cell_value(z,power_col)*projectsheet.cell_value(z,turb_col)
                  indexpower_array[0][1][y][x] = indexpower_array[0][1][y][x] +
projectsheet.cell_value(z,turb_col)
                if projectsheet.cell_value(z,diam_col) > 0:
                  newdiameter_array[0][1][y][x] = newdiameter_array[0][1][y][x] +
projectsheet.cell_value(z,diam_col)*projectsheet.cell_value(z,turb_col)
                  indexdiam_array[0][1][y][x] = indexdiam_array[0][1][y][x] +
projectsheet.cell_value(z,turb_col)
        if stategroup == 0:
          if projectsheet.cell_value(z,prov_col) == group_array[0][0][w]:
            if group_array[0][1][w] == group_list[0,x]:
              if projectsheet.cell_value(z,year_col) != '':
```

```
                y = projectsheet.cell_value(z,year_col) - 1980
                newprojects_array[0][1][y][x] = newprojects_array[0][1][y][x] + 1
                # print projectsheet.cell_value(z,year_col)
                if (projectsheet.cell_value(z,year_col) != '') &
(projectsheet.cell_value(z,year_col) <= 2015):
                  if projectsheet.cell_value(z,cap_col) > 0:
                    newcapacity_array[0][1][y][x] = newcapacity_array[0][1][y][x] +
projectsheet.cell_value(z,cap_col)
                  if projectsheet.cell_value(z,turb_col) > 0:
                    newturbines_array[0][1][y][x] = newturbines_array[0][1][y][x] +
projectsheet.cell_value(z,turb_col)
                    if projectsheet.cell_value(z,power_col) > 0:
                      newpower_array[0][1][y][x] = newpower_array[0][1][y][x] +
projectsheet.cell_value(z,power_col)*projectsheet.cell_value(z,turb_col)
                      indexpower_array[0][1][y][x] = indexpower_array[0][1][y][x] +
projectsheet.cell_value(z,turb_col)
                    if projectsheet.cell_value(z,diam_col) > 0:
                      newdiameter_array[0][1][y][x] = newdiameter_array[0][1][y][x] +
projectsheet.cell_value(z,diam_col)*projectsheet.cell_value(z,turb_col)
                      indexdiam_array[0][1][y][x] = indexdiam_array[0][1][y][x] +
projectsheet.cell_value(z,turb_col)

  years = 35 # need to set if number of years is updated for data sources
  turbine_array = NP.zeros(35)
  for x in range(0, 35):
    turbine_array[x] = totalsheet.cell_value(x+1,1)

  # now print the historical turbine and project data to the create the subscript
functions for VENSIM
  count = 0
  startyr = 1980
  f.write('\n\n')
  for x in range(0,num_groups):
    trigger = 0
    f.write('Historical Turbine Construction Rate[')
    f.write(newturbines_array[0][0][x])
    f.write(']([(0,0)-(2015,10)]')
    for y in range(0, col_end - col_start):
      f.write(', (')
      f.write(str(startyr + y))
      f.write(', ')
      f.write(str(newturbines_array[0][1][y][x]))
      f.write(')')
      trigger = trigger + 1
      if OP.mod(trigger,10) == 0:
        f.write('\\\n')
    f.write(') ')
    if count < (num_groups - 1):
      f.write('~~|\n')
    count = count + 1
  f.write('\n\t~\t')
  f.write('turbines/Year\n')
  f.write('\t~\tLookup function for turbine construction rate in a given state by
year.\t|\n')
```

```
  # # #
  count = 0
  startyr = 1980
  f.write('\n\n')
  for x in range(0,num_groups):
    trigger = 0
    f.write('Historical Turbine Installed Base[')
    f.write(newturbines_array[0][0][x])
    f.write(']([(0,0)-(2015,10)]')
    turbinesIB = 0
    for y in range(0, col_end - col_start):
      f.write(', (')
      f.write(str(startyr + y))
      f.write(', ')
      turbinesIB = turbinesIB + newturbines_array[0][1][y][x]
      f.write(str(turbinesIB))
      f.write(')')
      trigger = trigger + 1
      if OP.mod(trigger,10) == 0:
        f.write('\\\n')
    f.write(') ')
    if count < (num_groups - 1):
      f.write('~~|\n')
    count = count + 1
  f.write('\n\t~\t')
  f.write('turbines\n')
  f.write('\t~\tLookup function for total turbines installed in a given state by
year.\t|\n')

  # # #
  count = 0
  startyr = 1980
  f.write('\n\n')
  for x in range(0,num_groups):
    trigger = 0
    f.write('Historical Capacity Installed Base[')
    f.write(newcapacity_array[0][0][x])
    f.write(']([(0,0)-(2015,10)]')
    capacityIB = 0
    for y in range(0, col_end - col_start):
      f.write(', (')
      f.write(str(startyr + y))
      f.write(', ')
      capacityIB = capacityIB + newcapacity_array[0][1][y][x]/(1e6)
      f.write(str(capacityIB))
      f.write(')')
      trigger = trigger + 1
      if OP.mod(trigger,10) == 0:
        f.write('\\\n')
    f.write(') ')
    if count < (num_groups - 1):
      f.write('~~|\n')
    count = count + 1
  f.write('\n\t~\t')
  f.write('GW\n')
```

```
  f.write('\t~\tLookup function for total installed capacity in a given state by
year.\t|\n')

  # # #
  count = 0
  startyr = 1980
  f.write('\n\n')
  for x in range(0,num_groups):
    trigger = 0
    f.write('Historical Capacity Construction Rate[')
    f.write(newcapacity_array[0][0][x])
    f.write('](([(0,0)-(2015,10)]')
    for y in range(0, col_end - col_start):
      f.write(', (')
      f.write(str(startyr + y))
      f.write(', ')
      f.write(str(newcapacity_array[0][1][y][x]/(1e6)))
      f.write(')')
      trigger = trigger + 1
      if OP.mod(trigger,10) == 0:
        f.write('\\\n')
    f.write(') ')
    if count < (num_groups - 1):
      f.write('~~|\n')
    count = count + 1
  f.write('\n\t~\t')
  f.write('GW/Year\n')
  f.write('\t~\tLookup function for capacity construction rate in a given state by
year.\t|\n')

  # # #
  startyr = 1980
  f.write('\n\n')
  f.write('Historical rest of world turbine construction start rate')
  f.write('([(0,0)-(2015,10)]')
  for y in range(0,35):
    trigger = 0
    turbineexperience = 0
    for x in range(0, num_groups):
      turbineexperience = turbineexperience + newturbines_array[0][1][y][x]
    f.write(', (')
    f.write(str(startyr + y))
    f.write(', ')
    f.write(str(turbine_array[y]-turbineexperience))
    f.write(')')
    trigger = trigger + 1
    if OP.mod(trigger,10) == 0:
      f.write('\\\n')
  f.write(') ')
  f.write('\n\t~\t')
  f.write('turbines/year\n')
  f.write('\t~\tLookup function for turbine construction rate for the rest of the
world not included in the State group set by year.\t|\n')
```

```
# final action close the file with all the input in order
f.close # last action of script is to close the open file that exists
```