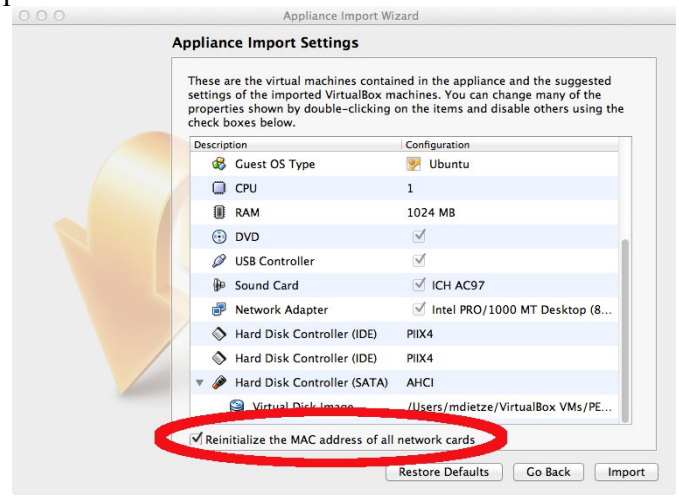
PEcAn v1.3.4 Hands-On Demo

Installing PEcAn:

1. PEcAn consists of a set of scripts and code that is compiled within a Linux operating system and saved in a "virtual machine (VM)". Virtual machines allow for running consistent set-ups without worrying about differences between operating systems, library dependencies, compiling the code, etc.
2. To run the PEcAn VM you will need to install VirtualBox, the program that runs the virtual machine (http://www.virtualbox.org). On Windows you may see a warning about Logo testing, it is okay to ignore the warning.
3. After you have Virtual Box installed you'll need to download the PEcAn virtual machine: http://isda.ncsa.illinois.edu/download/index.php?project=PEcAn. The virtual machine is available under the tag vm. Download the 32 bit version and note that the download is ~3 GB so will take from several minutes to hours depending on the connection speed.
4. To open up the virtual machine you'll first want to open up VirtualBox
5. The first time you use the VM you'll want to use File → Import Appliance in VirtualBox in order to import the VM. This will create a virtual machine from the disk image. When asked about the Appliance Import Settings **make sure you select "Reinitialize the MAC address of all network cards"**. This is not selected by default and can result in networking issues since multiple machines might claim to have the same network MAC Address. That said, users who have experienced network connection difficulties within the VM have sometimes had better luck after reinstalling without reinitializing.
6. Next, click "Import". You only have to do this Import step once, in the future you can just skip to the next step.

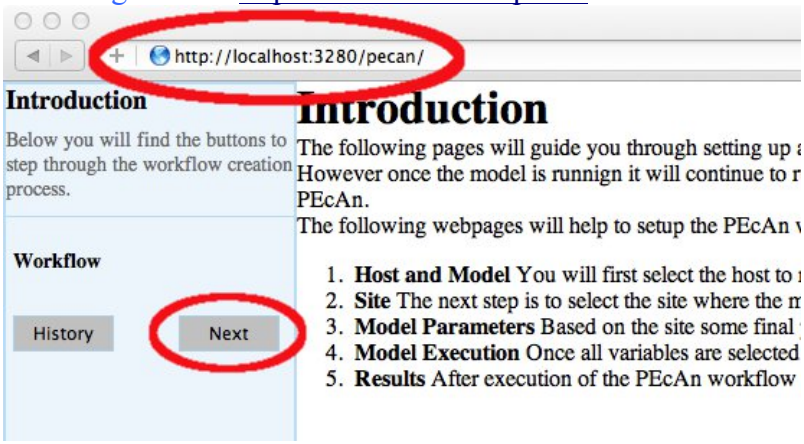


7. Start Pecan by double clicking on the icon for the VM. A terminal window will pop up showing the machine booting up which may take a minute. It is done booting when you get to the "pecan32 login:" prompt. You do not need to login as the VM behaves like a server that we will be accessing through you web browser. Feel free to minimize the VM window.

DEMO 1: Web-based tools

We will begin by exploring a set of web-based tools that are designed to run single-site model runs.

1. To access the web pages you can use a web browser on your own computer's desktop and connect to the following address http://localhost:3280/pecan



2. There are three steps to running PeCAN: selecting a site, specifying drivers, and running the model, as detailed below.

3. Site Selection
   1. From the Introduction page, click the 'Next' button to move to the "Select Host" page.
   2. PEcAn is designed to conduct runs both on the local VM and on remote machines. By selecting different servers from the "HOST" pull down menu you will be able to see what sites exist within the PEcAn databases on different servers. As the database also records sites where data was collected, not all sites will have the necessary driver files to run the models. For this exercise you will want to select the local machine "pecan32" to see the sites that are stored locally on the VM and have driver data.
   3. Next, you'll want to select the ecosystem model you want to run. We'll begin by using SIPNET because it is considerably faster than ED2.
   4. Finally, you'll want to select the site where you want to run the model. Feel free to zoom out to examine sites outside the initial frame, or to zoom in to examine clusters of sites or to see exactly where a site is located. When you click on a site's flag on the map, it will give you the name and location of the site. It will also put that site in the "Site:" box on the left hand site, which indicated its your current site selection. For this tutorial, choose the Sylvania site as the site for your run.
   5. Once you have selected a Host, Site, and Model, click "Next"

4. Run Specification
   Next we will select the input files required to run the model and the vegetation type.
   1. The **Weather Data file** pull-down will allow you to select the meteorological driver dataset that you want to use by name and time span. Most of the sites will only have a single option.
   2. You'll want to select the Plant Functional Type that you want to run from the **PFT** menu. While ecosystem models do have broad similarities, there is still a degree of model-specificity in model parameters, thus the list of available PFTs will vary depending upon which model you selected. Some models, such as ED2, allow for the competition between

PFTs within a site and thus selecting multiple PFTs is allowed (and in fact encouraged). SIPNET currently only runs one PFT at a time so you will want to only select a single PFT

3. Finally, click **Next** to start the model run.

5. Model Run
   SIPNET will execute the model run fast enough that you are liable to miss some of the following steps (it's more apparent with ED2), but the PEcAn workflow will automatically run the following, indicating model progress at each step
   1. fia2ed: reads forest inventory data (currently only runs if ED2 is selected)
   2. query.trait: reads trait data
   3. meta.analysis: synthesizes trait data to estimate model parameters
   4. write.config: writes model settings and parameter files
   5. model: run model
   6. finished: clean up

6. Congratulations! If you got this far, you have managed to run an ecosystem model without ever touching a line of code! Now it's time to look at the results.

7. Output and Visualization

   1. **Plots**
      1. Select a **Year** and output **Variable**, click 'Plot run/year/variable'
      2. Within this figure the **points indicate the daily mean** for the variable while the **envelope encompasses the diurnal variability** (max and min).
      3. Units, variable names, and descriptions are all drawn automatically from the meta-data in a standardized, model-independent netCDF output format (derived from NACP, LBA, PalEON, and MsTMIP model-data inter-comparison projects). See http://nacp.ornl.gov/MsTMIP_variables.shtml for variable names and units.
      4. Figures are *.png files that can be saved by right-clicking and selecting "Save Image As...". However, as will be described below it is easy to go back to past runs in PEcAn so you don't have to download figures or files in order to keep a record of your results.
      5. Try looking at a number of different output variables over different years.

   2. **Outputs:**
      A number of files generated by the underlying ecosystem model are archived and available for download. are archived and available for download. These include:
      1. A summary file (README.txt),
      2. Output files in the standardized netCDF ([year].nc)
      3. Raw model output in model-specific format (e.g. sipnet.out).

      For example, one could easily download model output and plot model versus data in Matlab or R, or visualize netCDF output in ncview.

   3. **PFT Files:**
      There is a wide array of outputs available that are related to the process of estimating the model parameters for a specific Plant Functional Type. We will revisit many of these files in Demo 2 when we look at the meta-analysis in more detail, but the following is the general idea of what is contained in different files:
      1. *.bug files provide the statistical code used to estimate each parameter in JAGS.
      2. The ma.summaryplots.*.pdf are collections of diagnostic plots produced in R after the

above JAGS code was run that are useful in assessing whether the statistical model converged.

3. The Rdata files trait.data.Rdata and madata.Rdata are, respectively, the available trait data extracted from the database that was used to estimate the model parameters and that same data cleaned and formatted for the statistical code.
4. The file traits.mcmc.Rdata contains the raw output from the statistical code. This includes samples from all of the parameters in the meta-analysis model, not just those that feed forward to the ecosystem, but also the variances, fixed effects, and random effects.
5. The files prior.distns.Rdata and post.distns.Rdata store simple tables of the prior and posterior distributions for all model parameters in terms of the name of the distribution and its parameters.
6. Finally, posteriors.pdf provides graphics showing, for each model parameter, the prior distribution, the data, the smoothed histogram of the posterior distribution (labeled *post*), and the best-fit analytical approximation to that smoothed histogram (labeled *approx*)

4. **PEcAn Files:**
   Likewise, a number log files from the PEcAn workflow are available from the execution of the workflow. These include:
   1. The PEcAn settings file (pecan.xml),
   2. The workflow scripts (workflow_stage[1,2,3**]**.R),
   3. The corresponding log files (workflow_stage[1,2,3**]**.Rout), and
   4. The Monte Carlo samples used in the runs (samples.Rdata).

   These files are primarily useful if the workflow generates an error, and as metadata (a detailed record of how data was generated), but also includes useful diagnostics from the meta-analysis. If, by chance, you get an error in your run today it is probably simplest to go back and try changing options and re-running (e.g. with a different site or PFT), as time does not permit detailed debugging.

8. Model Run Archive
   1. Click on the HISTORY button at the start or end of the workflow to go to a table recording past model runs. Click on any previous run in the "ID" column to go to the current state of that run's execution. This is most commonly the output and visualization page, but if you are doing a long run you can always return to that run in-progress this way. The run you just did should be the more recent entry in the table. We encourage you in your own time to run the ED2 model for your site for comparison and because ED2 also has a more diverse range of output variables to explore.

DEMO 2: PEcAn R Modules

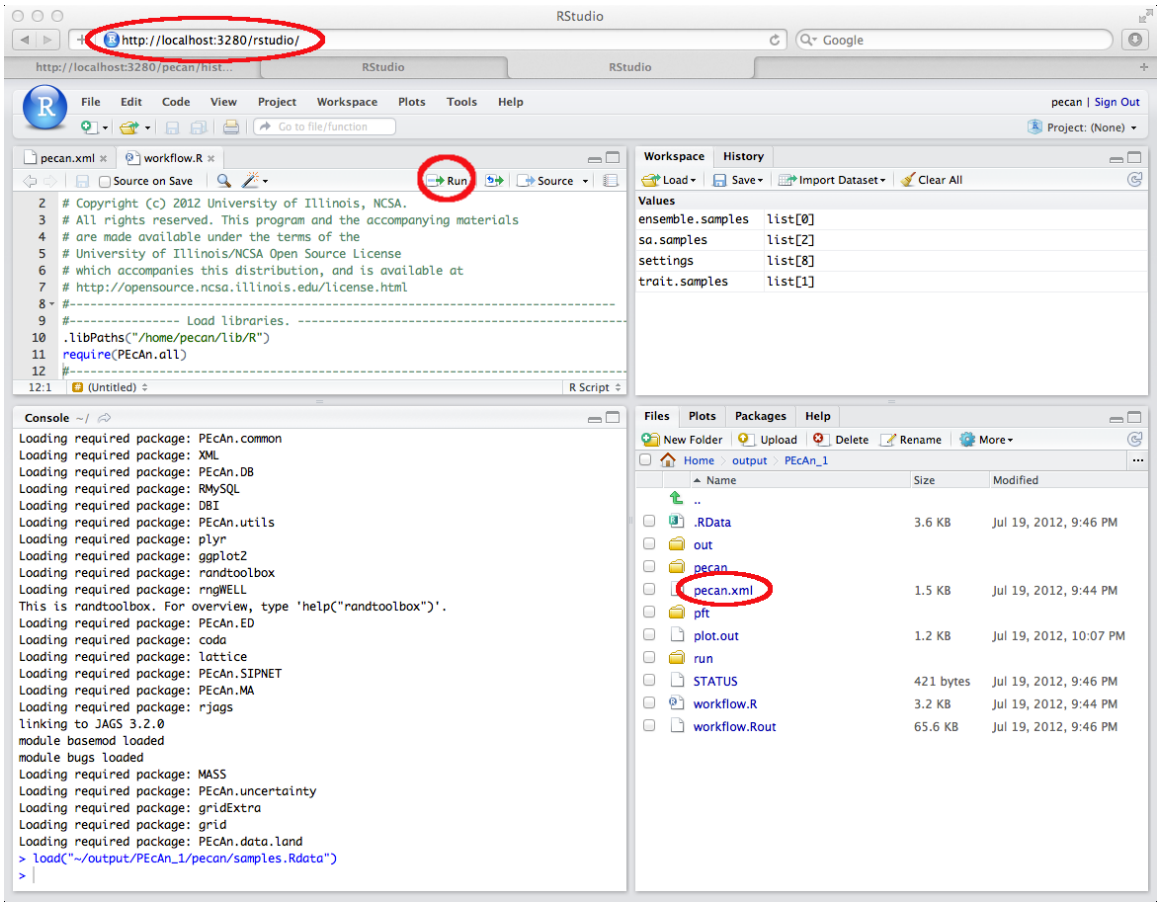
Currently, the model's web interface is limited to the setup, running, and visualization of a single run at a single site. To explore PEcAn's ensemble run, sensitivity analysis, and variance decomposition functionality we will need to work in the statistical programming language R. To make this easier we have installed a brower-based R interface, RStudio Server (http://rstudio.org), on the VM.

1. Open RStudio Server:

    1. Open up a second window on your web browser to http://localhost:3280/rstudio
    If you closed your previous PEcAn session, make sure the first window is open and points to http://localhost:3280/pecan

    2. The username for the VM is *carya* and the password is *illinois*.

    3. You do not need to do this now, but FYI it is also possible to ssh into the virtual machine using the following command in terminal:
    ssh -l pecan -p 3222 localhost



2. Open PEcAn settings file

    1. Go back to http://localhost:3280/pecan and from the "History" page on the PEcAn web

interface note the ID for your model run.

2. Login to the Virtual Machine terminal
    1. Create a new directory for your runs
       mkdir demo

    2. Copy a "cleaned" version of the PEcAn settings file from your previous run to your new directory, **where # is your run ID**.
       pecan/scripts/cleansettings.R output/PEcAn_#/pecan.xml demo/demo.xml

3. In the RStudio window, navigate to the Home > demo folder in the file browser located in the lower right corner of the screen.

4. Click on **demo.xml** to open the file. This file records the settings used by PEcAn to produce your run in a standardize format known as XML, which is quite similar to the HTML language used to design webpages.

3. Modify settings to run a meta-analysis, sensitivity analysis, and ensemble analysis

    1. First, we are going to need to change the location of the output directories so that the output of these new runs goes to a new location. Find the tag labeled <outdir> and change the contents of the tag to the location of the folder that you want output written to (e.g. <outdir>**/home/carya/demo/**</outdir>). Note that since our user id is 'carya", the "Home" we see in RStudio is equivalent to /home/carya/ in the underlying linux OS.

    2. Next, we're going to modify two settings in the meta-analysis.
        1. In order to run the meta.analysis, within <meta.analysis> set
           <update>TRUE</update>.

    3. In order to run an ensemble analysis, we need to tell PEcAn to turn on this module and let it know how many runs we want in our ensemble. We do this by changing the value set in the ensemble tag from 1 run to 50 runs so that it looks like.:

```
<ensemble>
        <size>50</size>
</ensemble>
```

       As a reminder, the ensemble analysis will sample all model parameters randomly from their posterior distributions in order to propagate this uncertainty into the model output.

    4. In order to run the sensitivity analysis, we likewise need to turn on this module, and tell PEcAn how we want the parameters varied. PEcAn's sensitivity analysis holds all parameters at their median value and then varies each parameter one-at-a-time based on the *quantiles* of the posterior distribution. For example, to sample at the edge of a 95% CI you would use the <quantile> tag to set the quantiles to 0.025 and 0.975. PEcAn also includes a handy shortcut, the <sigma> tag, that converts a specified standard deviation into its Normal quantile equivalent. For example, 1 and -1 are converted to 0.157 and 0.841 (a 68.2% CI). Finally, we need to specify what output variable we are calculating sensitivity for and what time period we are averaging this variable over. To specify time

you use the <start.year> and <end.year> tags. The <variable> tag is a character string that follows the output standards used by recent inter-comparison projects (MsTMIP, NACP, LBA, PalEON), see http://nacp.ornl.gov/MsTMIP_variables.shtml for variable names and units.
Below is an example for GPP that samples from -3 to 3 sigma using years 2002-2006. Copy and paste it into your sylvania.xml file right after the initial <pecan> tag:

```
<sensitivity.analysis>
        <quantiles>
                <sigma>-3</sigma>
                <sigma>-2</sigma>
                <sigma>-1</sigma>
                <sigma>1</sigma>
                <sigma>2</sigma>
                <sigma>3</sigma>
        </quantiles>
        <start.year>2002</start.year>
        <end.year>2005</end.year>
        <variable>NPP</variable>
</sensitivity.analysis>
```

Check that the variable, start.year, and end.year for the ensemble and sensitivity.analysis are the same. If one set of these tags is missing, the system will assume that they are the same and use the information from one to fill in the other.

5. Within this file you will also see other settings for different modules. Hopefully the naming of these is intuitive. The latest version of the pecan.xml documentation, and further documentation of the PEcAn system in general, can be found in the wiki section of our Github website (https://github.com/PecanProject/pecan/wiki/PEcAn-Configuration)

6. Save you changes

ERRORS: The PEcAn code will return a lot of *information* in red font when commands are run. Most of these, including items labeled as warnings, INFO, and DEBUG, are not errors. In general, errors are explicitly labeled with the word ERROR

4. Initialize PEcAn Workflow

    1. In the **Files** window, go to the **Home > pecan > scripts** directory

    2. Open the file **workflow.R**. This script provides an example of some of the high-level modules within PEcAn and we will use it to run the meta-analysis, sensitivity analysis, ensemble analysis, and variance decomposition.

    3. The modules we are using are all written in R and saved as a set of R packages, which are link together in the PEcAn.all package. To load this package select all lines up to the following line and click “run”:

```
require(PEcAn.all)
```

4. The next line of code will load the settings file we just made. Before running it you will need to fill in the path of your settings file. For example:
   ```
   settings <- read.settings("/home/carya/demo/demo.xml")
   ```
   If you type 'settings' at the command line in the **Console** window you will see the full contents of the settings file as an R list object

5. Run `unlink(file.path(settings$outdir, "STATUS"))` to reset status checking. Once you do this the status of each of the steps, and all the output, should be visible through the History page on the PEcAn web interface http://localhost:3280/pecan/history.php

6. Next, `get.trait.data()` queries the database for both the trait data and prior distributions associated with the PFTs specified in the settings file. For this and the next few functions you will need to **run the status.start and status.end before each function**. The list of variables that are queried is determined by what variables have priors associated with them in the definition of the pft. Likewise, the list of species that are associated with a PFT determines what subset of data is extracted out of all data matching a given variable name. Demo 3 will demonstrate how a PFT can be created or modified.
   The output is written to two Rdata files, **trait.data.Rdata** and **prior.distns.Rdata**, in the location specified by `<pft><outdir>` in the settings. Navigate to that location in the file browser and click on these files to load them into your workspace. You can further examine them in the **Workspace** window or accessing them at the command line. For example, try typing `names(trait.data)` as this will tell you what variables were extracted, `names(trait.data$Amax)` will tell you the names of the columns in the Amax table, and `summary(trait.data$Amax)` will give you summary data about the Amax values.

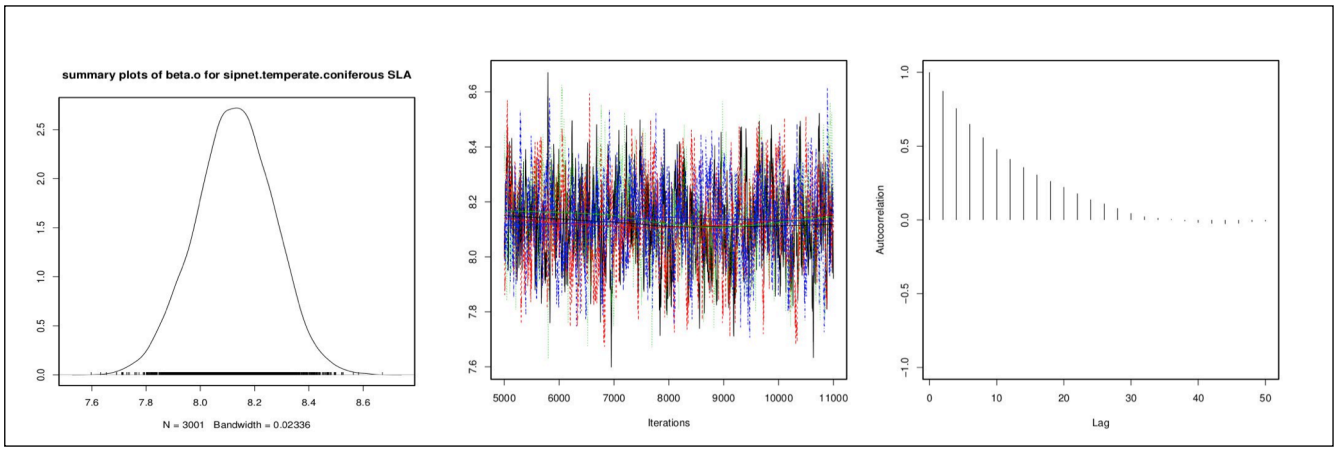
5. Meta-analysis: `run.meta.analysis()`.
   The following points is explain how this code works and the files it produces. All files can be found in the set output folder.

   1. The meta-analysis code begins by distilling the trait.data to just the values needed for the meta-analysis statistical model, with this being stored in `madata.Rdata`. This reduced form includes the conversion of all error statistics into precision (1/variance), and the indexing of sites, treatments, and greenhouse. In reality, the core meta-analysis code can be run independent of the trait database as long as input data is correctly formatted into the form shown in madata.

   2. The evaluation of the meta-analysis is done using a Bayesian statistical software package called JAGS that is called by the R code. For each trait, the R code will generate a `[trait].model.bug` file that is the JAGS code for the meta-analysis itself. This code is generated on the fly, with PEcAn adding or subtracting the site, treatment, and greenhouse terms depending upon the presence of these effects in the data itself. If the `<random.effects>` tag is set to FALSE then all random effects will be turned off even if there are multiple sites and treatments.

3. The code then calls JAGS to fit the model itself and you will see a series of *** as a progress bar on the screen as the sampler progresses through the <iter> iteration of the Markov Chain/Monte Carlo (MCMC) sampler.

4. When the MCMC finishes it will output a number of diagnostics, including the summary statistics of the model, an assessment of whether the posterior is consistent with the prior, and the status of the Gelman-Brooks-Rubin convergence statistic (which is ideally 1.0 but should be less than 1.1). If you missed any of these as they scrolled by all of this output can be found in the meta-analysis.log file.



5. Next, the code generates a number of diagnostic figures in a ma.summaryplots.[pft].[trait].pdf plot for each pft and trait. Open up one of these pdfs in the file browser to evaluate the shape of the posterior distributions (they should generally be unimodal), the convergence of the MCMC chains (all chains should be mixing well from the same distribution), and the autocorrelation of the samples (should be low).

6. Finally, the code attempts to fit an analytical approximation to the MCMC posterior. The outputs of this are post.distns.Rdata, which is structured identically to the prior.distns but updated for the meta-analysis, and posteriors.pdf, which is a series of graphs comparing the MCMC posterior, the analytical approximation, the data, and the priors. Open posteriors.pdf and compare the posteriors to the priors and data.

6. Sensitivity Analysis and Ensemble Analysis
These two analyses are run together by a number of function calls:
   1. run.write.configs(model,write)
      This code takes all the parameters and drivers in PEcAn and writes out a series of model-specific settings files for each of the runs required for the sensitivity and ensemble analyses.
      1. This function will store the parameter setting used in samples.Rdata. This file contains two data objects, sa.samples and ensemble.samples, that are the parameter values for the sensitivity analysis and ensemble runs respectively.
      2. There is a model-specific interface module, write.config.[model], that has to be implemented for each model running in PEcAn. The output from this function is obviously model-specific, but for SIPNET you will see your output directory

populated with a number of subdirectories, one for each model run, which contain the following files: [run.id].clim, [run.id].param, [run.id].param-spatial.

3. There is also a met2model.[model] module that converts the standard netCDF met file in PEcAn into the model-specific driver format.

2. start.model.runs(model)
   This code starts the model runs using a model specific run function named start.runs.[model]. If the ecosystem model is running on a remote server, this module also takes care of all of the communication with the remote server and its run queue. Each of your subdirectories should now have a [run.id].out file in it. This step will take a while to run.

3. convert.outputs(model,settings)
   get.model.output(model,settings)
   The first function uses a model-specific model2netcdf.[model] function to convert the model output into the standard output format (see http://nacp.ornl.gov/MsTMIP_variables.shtml).

   The second function extracts the data for the requested variables, averages over the time-period requested, and store the output in the output.Rdata file. This file contains two objects, the sensitivity.output and the ensemble.output, that is the model prediction at the parameter values given in sa.samples and ensemble.samples. In order to save bandwidth, if the model output is stored on a remote system (e.g. a computer cluster or in the cloud) PEcAn will perform these operations on the remote host and only return the output.Rdata object.

   This step will take a while to run and you will want to run all the code between the status.start and status.end.

4. get.model.output(model,settings)

   Interactive

5. run.ensemble.analysis()
   run.ensemble.analysis(plot.timeseries = TRUE)
   This module makes some simple graphs of the ensemble output. Open ensemble.analysis.pdf to view the ensemble prediction as both a histogram and a boxplot. Open ensemble.ts.pdf to view a timeseries plot of the ensemble mean, meadian, and 95% CI

6. run.sensitivity.analysis()
   This function processes the output of the previous module into sensitivity analysis plots, **sensitivityanalysis.pdf**, and a variance decomposition plot, **variancedecomposition.pdf**, which you can open in your web browser by clicking on them in your Rstudio Files window. In the sensitivity plots you will see the parameter values on the x-axis, the model output on the Y, with the dots being the model evaluations and the line being the spline fit. The variance decomposition plot is discussed more below. For your reference, the R list object, sensitivity.results, stored in

sensitivity.results.Rdata, contains all the components of the variance decomposition table, as well as the input parameter space and splines from the sensitivity analysis (reminder: the output parameter space from the sensitivity analysis was in outputs.R).

7. Variance Decomposition and model-data feedbacks
   The variance decomposition plot contains three columns, the coefficient of variation (normalized posterior variance), the elasticity (normalized sensitivity), and the partial standard deviation of each model parameter.

   QUESTIONS:
   1. This graph is sorted by the variable explaining the largest amount of variability in the model output (right hand column). From this graph identify the top-tier parameters that you would target for future constraint.
   2. A parameter can be important because it is highly sensitive, because it is highly uncertain, or both. Identify parameters in your output that meet each of these criteria. Additionally, identify parameters that are highly uncertain but unimportant (due to low sensitivity) and those that are highly sensitive but unimportant (due to low uncertainty).
   3. Parameter constraints could come from further literature synthesis, from direct measurement of the trait, or from data assimilation. Choose the parameter that you think provides the most efficient means of reducing model uncertainty and propose how you might best reduce uncertainty in this process. In making this choice remember that not all processes in models can be directly observed, and that the cost-per-sample for different measurements can vary tremendously (and thus the parameter you measure next is not always the one contributing the most to model variability). Also consider the role of parameter uncertainty versus model sensitivity in justifying your choice of what parameters to constrain.

Demo 3: bonus material [as time permits]

For these activities, make sure to copy your settings file over to a new location, and change the <outdir>, <rundir>, etc. so that you don't write over your previous runs.

1. Re-run the sensitivity analysis and variance decomposition for a different output variable, time period, site, or PFT and compare results with first site.
    1. Make sure to send the output to a DIFFERENT directory so that your original work is not written over.
    2. If you are only changing the output variable or time period, you don't need to re-run the whole workflow since all the output variables are in the model output files. You can modify the settings object within R and run the workflow from the extraction of the output from the model runs onward (from get.model.output on)
    3. If you are running a new site, it is generally a good idea to do a test run first on http://localhost:3280/pecan to make sure the model behaves sensibly before conducting hundreds of (potentially uninformative) runs.
2. Use the web database interface to change priors, define a new site, or define a new PFT
    1. Open up a new browser window to http://localhost:3280/bety . This page is the web-interface to the database underlying PEcAn – it started as an independent project as part of our biofuel work, hence the name BETY-db, but is now much more generic (apologies for the fact that web design has not kept up). For those that are interested the full version of the database is at betydb.org.
    2. To modify or add a new Plant Functional Type (PFT), or to change a PFT's priors, navigate on the grey menu bar to Data > PFTs
        1. To add a new pft, click “new PFT” at the top and enter a name and description. (hint: we're trying to name PFTs based on model.biome.pft, ED2 is the default model if one isn't specified)
        2. To add new species to a PFT click on [+] View Related Species and type the species, genus, or family you are looking for into the Search box. Click on the + to add.
        3. To remove a species from a PFT, click on [+] View Related Species and click on the X of the species you want to remove from the PFT.
        4. To remove a prior, click [-] View Related Prior and click on the X of the variable who's prior you want to remove. This will cause the parameter to be excluded from all analyses (meta-analysis, sensitivity analysis, etc) and revert to its default value.
        5. To add a prior, choose one from the white box of priors on the right to choose.
        6. To view the specification of a prior, or to add a new prior, click BETY-DB > Priors and enter the information on the variable, distribution name, distribution parameters, etc. N is the sample size underlying the prior specification (0 is ok for uninformative priors).
        7. You can also got to Data > Variables in order to use the search function to find an existing variable (or create a new one). Please try not to create new variables unnecessarily (e.g. changes of variable name or units to what your model uses is handled internally, so you want to find the trait with the correct MEANING).
    3. To add a new site
        1. The database currently requires that a site be associated with a citation. Navigate to > CITATIONS and select a reference for the study site by clicking the check mark, or create a new citation (creating a 'pers. communication' or 'unpublished data' citation IS allowed)
        2. Click on the **New Site** button. Fill out at least a name and lat/lon (which can be done by

hand or by clicking on the map).

3. To upload model driver data [or any other input data] for the site (currently required in order to run the model at a site)
    1. Select RUNS > INPUTS
    2. At the bottom of the page click on New Input
    3. Choose your site from the pulldown menu
    4. Set the date range, file format, and other meta-data
    5. Click “Create”
    6. To associate the Input record in the database, click “Edit” and then “New File”
    7. Select the machine that the file is on (pecan32 is the VM itself) and fill in the directory path where the file is located and the file name
    8. Click Update