**[[BOTTOM]](http://pmatwiki.sandia.gov/pmatwiki/how-to-use-eCo-Pylot?action=print#bottom)[[TOP]](http://pmatwiki.sandia.gov/pmatwiki/how-to-use-eCo-Pylot?action=print#top)How to use eCo-Pylot**

**[[BOTTOM]](http://pmatwiki.sandia.gov/pmatwiki/how-to-use-eCo-Pylot?action=print#bottom)[[TOP]](http://pmatwiki.sandia.gov/pmatwiki/how-to-use-eCo-Pylot?action=print#top)Introduction**

eCo-Pylot is part of the code triumvirate of [Pylot](http://pmatwiki.sandia.gov/pmatwiki/how-to-use-pylot) / [Co-Pylot](http://pmatwiki.sandia.gov/pmatwiki/how-to-use-Co-Pylot) / eCo-Pylot. These codes can be used in stand-alone mode or together to easily move data files to database tables on a remote server (Co-Pylot, eCo-Pylot) as well as display the tables in an intuitive interface and analyze table data using various kinds of plots and statistical analysis (Pylot).

eCo-Pylot is not a graphical interface like Co-Pylot. Rather, eCo-Pylot is a script written in Python that resides on Sandia's database server *face.sandia.gov*. In general, the script is designed to intercept emails sent to a specific address, extract one or more attached datafiles, parse the email for user and database information, and insert the information into a database table. Each attached datafile inserts into separate table rows. The target database is specified in the username. The target database table is specified in the subject line. The body of the email is inserted as user comments in the database table. Finally, an email is sent to the user indicating whether the contents sent by the user were successfully inserted into the designated table.

Transferring files to a database via email represents an extremely easy and intuitive method for moving data into a database. In addition, if the database server is public (not the case at Sandia), files can be sent to a particular database from anywhere in the world, given an email client.

Though easy and intuitive, eCo-Pylot does not capture as much information as the Co-Pylot interface (*e.g.*, directory and machine location of the output, input, and executable files). Users should compare descriptions between eCo-Pylot and Co-Pylot to determine which approach best fits their needs.

Suggestions for improvements to the script are welcome.

**[[BOTTOM]](http://pmatwiki.sandia.gov/pmatwiki/how-to-use-eCo-Pylot?action=print#bottom)[[TOP]](http://pmatwiki.sandia.gov/pmatwiki/how-to-use-eCo-Pylot?action=print#top)Sending Email Through eCo-Pylot**

Sending data to a database table *via* eCo-Pylot is as simple as writing an email. An example will best show how this is done.

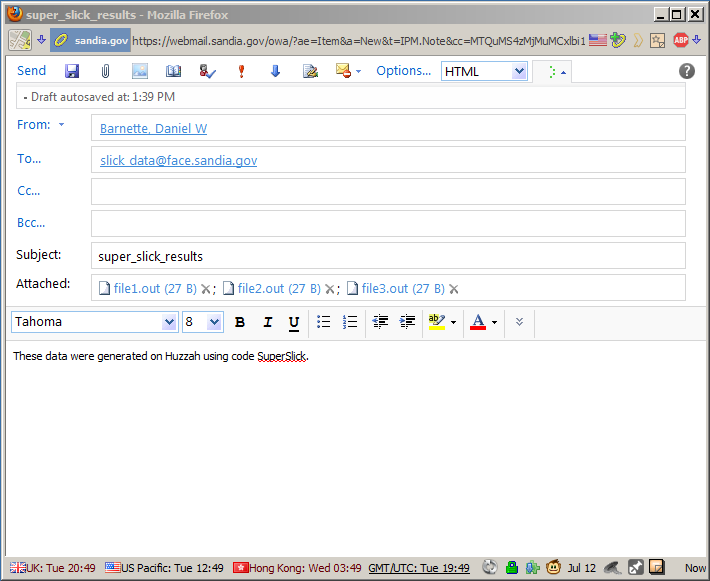
Assume the following exists on a remote database server accessible by the eCo-Pylot script:

* Database server: face.sandia.gov
* Database name: slick\_data
* Database table name: super\_slick\_results
* Database username: slick
* Database password: data

Also assume the user has generated 3 data files and wishes to send a user comment:

* Data files to send to the database table: file1.out, file2.out, file3.out
* User comment: 'These data were generated on Huzzah using code SuperSlick.'

The email to be sent now looks as follows:



Note that

* access to the database server is *via* the username and password extracted from the database name
* the database name and the server location are both specified in the **To...** field in the form *database\_name@server\_location*
* the database table name is specified in the **Subject:** field
* the datafiles are sent as attachments; the files are converted by the email application into encoded base64 files before being attached; eCo-Pylot decodes the datafiles before inserting them one row per file in the database table
* the body of the email is inserted into a database field typically labeled as *user\_comment*.

Also extracted or calculated from the email are

* the sender's email address; used by eCo-Pylot for responding back to the user whether the database insertion was successful
* date and time the email was sent
* which machine the email was sent from
* sender's name
* day of week

If the database insertion is not successful, eCo-Pylot responds to the user with an error message explaining the reason for the failure. The reasons are typically minor and easily correctable, like leaving out the database table name from the **Subject:** line, for example.

The targeted database table is unaffected if the insertion is not successful.

**[[BOTTOM]](http://pmatwiki.sandia.gov/pmatwiki/how-to-use-eCo-Pylot?action=print#bottom)[[TOP]](http://pmatwiki.sandia.gov/pmatwiki/how-to-use-eCo-Pylot?action=print#top)Behind the Scenes with eCo-Pylot**

What follows is a more detailed discussion of how eCo-Pylot is implemented on the MySQL server *face.sandia.gov*.

First, the user sends an email with attached files to an address such as, for example, [*path2exascale\_data@face.sandia.gov*](mailto:path2exascale_data@face.sandia.gov). The email is sent to the server *face.sandia.gov* as any email with this domain name would be. It is directed to the user *path2exascale\_data* as specified in the email address. This user account has previously been set up on *face.sandia.gov*, which is where eCo-Pylot runs.

A *.forward* file intercepts the email and sends a copy to the typical email directory on a linux machine */var/spool/mail*. The *.forward* file also sends a copy of the email to the script *parse\_mail* that runs eCo-Pylot.

The *.forward* file that would be used on the server *face.sandia.gov* for user *path2exascale\_data* is

path2exascale\_data, "| /home/dwbarne/Parser/parse\_mail"

The *parse\_mail* script is given by

cat "$@" | /usr/bin/python /home/dwbarne/Parser/eco\_pylot.py 1> output\_eco\_pylot.txt 2> errors\_eco\_pylot.txt

The Python program *eco\_pylot.py* reads the email *via* standard input and begins parsing the email, as discussed above. *eco-pylot.py* then sends the parsed data to the database named in the email address, *path2exascale\_data*. The database server is setup to take *path2exascale* as the database server login username and *data* as the database server login password.

In summary, we specify the database name (*path2exascale\_data*), the database server (*face.sandia.gov*), the server login username (*path2exascale*), and the server login password (*data*), all in the email address. The database table into which the data is inserted is specified in the **Subject:** field. Data files are simply attachments, one or many. User comments are specified in the body of the email.

Of course, the structure of the target database table must be identical to the format of the MySQL statement in *eco-pylot.py* whose job it is to insert the parsed data. If insertion is successful, a return email is sent to the user stating success. If for any reason the insertion is not successful, a return email is sent stating why, if possible. Only one email is sent back to the user, regardless of how many files were attached for insertion.