A LyX/Sweave Tutorial

Jeff Laake

May 15, 2012

1 Introduction

- What are LyX, LaTex, MikTeX, and Sweave?
  - LyX is an open source document processor running on Linux/Unix, Windows, and Mac OS X that creates LaTex documents.
  - LaTeX is a document markup language and document preparation system for the TeX typesetting program.
  - MiKTeX is a complete TeX system for Windows. TexShop and MacTex are two TeX implementations for Mac.
  - Sweave is an R package that provides a framework for mixing text and R code for automatic document generation. A single source file contains both documentation text and R code, which are then woven into a final complete document.
- Why use LyX?
  - LyX provides the user a graphical user interface and the familiar look of a WYSIWYG word processor for the quite complex LaTeX markup language. You can use the complex LaTeX language to make professional looking documents without memorizing or typing the markup language which many would find rather arcane and difficult to learn.
  - In addition to making LaTeX easier, LyX also automates the Sweave process. When you create a complete document (eg pdf) that contains R code, LyX converts the .lyx file to an .rnw file, processes the .rnw file with Sweave to insert the results of the R code to create a .tex file which is then processed by MikTex to create the final document (eg pdf).
  - Even if you don't use Sweave, LyX as an interface to a TeX system like MikTeX is a worthwhile endeavor to create professional looking documents without learning and typing a lot of LaTeX but still being able to use it when necessary.
- What problems you should expect
  - Collaboration: Many of your collaborators will want to use Word or some other word processors and may be hesitant to learn a new one. If you are the senior author then I suggest that you are the one in control unless the junior author is your boss. Regardless, a collaborator can install and use LyX to modify documents and track changes quite easily and it should function very much like Word and other gui word processors. You can close all of the Sweave and other Tex insets so it does not appear so daunting a foreign to your collaborators. That also lets them focus on the text.

- Publication outlets: This is a much more imporant issue and one that you should consider very seriously before you decide to adopt LaTeX and the Sweave approach for your primary method. Many journals simply refuse to accept anything but a Word document. Thus, if you want to submit to that journal you are committed to converting your document to Word once it is completed. There are a couple of options here. One is to use a piece of software like Tex2Word which takes the Sweaved output in the .tex file and converts it to Word .docx file. It is not perfect and I've found that it converts equations into images which is not optimal. The other option is a pdf to Word converter. So far I've had less luck with this than the Tex2Word conversion. All of these types of programs are improving and eventually this should be a big problem. Also, I've recently learned that there is an effort to develop software that will work with knitr to create pdfs and Word documents seamlessly but this is just an idea at present. There is also an R package called SWordInstaller that will install software that allows the Sweave approach to be used with Word. I tried and it appears to work but I don't have any experience with it yet but much of the Sweave structure described here appears to be used with SWord.
- Frustration: Oh yea, you will get frustrated as you do learning any new piece of software and LyX can be particularly frustrating with some arcane error messages you will get. The problem is not always LyX and sometimes it is difficult to assess where the problem lies because you have several tools interacting with LyX, R (Sweave) and MikTeX. I'll offer some ideas to reduce your level of frustration by showing you some tools to help locate the source of the error. Also, I recommend viewing your document frequently between changes especially as you are learning. Then if something doesn't work you can revert back to see what caused the error. When you get an error that is Tex related and not in Sweave, you have the option to view the complete log and jump between errors but I've not found that particularly useful. If it has been awhile since I previewed the document and I get an error, I copy the file and then delete portions of the text until the error disappears. If the error remains after deleting some text, I hit the undo button and try another portion until I locate the offensive code. This is a bit of a hack but it works for me. I used it for this document and discovered that I had a problem which has plagued me and you will likely run into it as well. For a reason unknown to me, on occasion the alignment of the floats, TeX boxes can cause an error. Typically it is because the enter key (CR) has been hit between floats/TeX boxes and one will appear indented in comparison to the other. Place the cursor on the left hand side of the indented box and hit delete and it should shift to the left and the problem should disappear if that was the portion of the document causing the problem. You also have the option of looking at the Tex file and sometimes this will help but my knowledge of Tex is fairly remedial.

- LyX menu overview
  - File▷ Import/Export for conversion of file types
  - Edit▷ Paste Special
  - View▷Open/close insets; view source; view messages
  - Insert▷ Used to insert different kinds of objects like tables/figures/TeX formatting
  - Navigate▷ quick way to get to different parts of the document
  - Document▷Change tracking; Settings: Document class, Module, Preamble, Fonts
  - Tools▷Reconfigure (copy problem; add features/classes); Preferences - Paths

  - Help ▷ Help file as a LyX file - convert to pdf; Sweave manual

- LyX toolbar overview
  - Environment choice box
  - Undo, Cut/Paste, Find and replace
  - Emphasis/Apply Last
  - Insert Equation/Graphic/Table - and Floats
  - View document
  - Lists
  - Font Changes/Paragraph settings

The help files, tutorial and examples (C:\Program Files (x86)\LyX20\Resources\examples) that accompany LyX are very complete and this is not an attempt to replace those. My focus here is on the steps you would take to create a manuscript that uses Sweave to produce a reproducible research document including text, equations, Sweave chunks for analysis, figures and tables with labelling and cross-referencing, and bibliography and citations. You can find much of this material scattered around in various places and this is an attempt to pull together much of that material in a single place for the benefit of myself and others. As with the LyX help manuals, this document can be used as a template for your own documents to enable you to mimic different aspects of the document that you want to create.

Some of the material I will cover is also covered in the LyX tutorial help file and the Sweave Manual written by Yihui Xie, Gregor Gorjanc and Jean-Marc Lasgouttes (Help ▷ Specific Manuals ▷ Sweave Manual). The original Sweave Manual written by Friedrich Leisch and R core members is available in the help for R (Help ▷ Manuals in PDF ▷ Sweave User) and is worth reading once you have learned the basics. I'm focussing here on the use of Sweave but I should mention a recent advance is the new R package knitr which can be used as a replacement for Sweave with many additional useful options. Once you learn the basics of using Sweave with LyX, conversion to knitr is quite easy (http://yihui.name/knitr/)

Also, you may want to get a good book on LaTeX. The book I have is A guide To LaTeX by Kopka and Daly; however it is rather old now and a more recent text may be more useful. You can also find the answers to most LaTeX questions with a quick search of the internet. Another useful place to look is the LyX wiki (http://wiki.lyx.org/) but beware that some information may be outdated as there is on-going development with LyX.

I should finish the introduction with an important caveat. I am somewhat of a beginner with LaTeX and LyX even though I have been using it for a couple of years now. There is much more to these tools than I know and there may be better ways of doing things than what I describe here. My expectation is that many of you will learn other tricks and ways of operating with LyX, LaTeX and Sweave. In the spirit of open source software, I encourage you to share freely with others.

2 Creating a new document

As with any word processor, creating a new document only requires selecting File ▷ New (Figure 1) which opens an empty document with a temporary name which you can change by using File ▷ Save As. Each LaTeX document has a document class and the default class is article which is what you will normally use but there are many document classes that can be used (e.g., book). You can change the Document class by selecting Document ▷ Settings from the LyX menu (Figure 1). To use Sweave, you need to add the Sweave module to the document as shown in Figure 2.

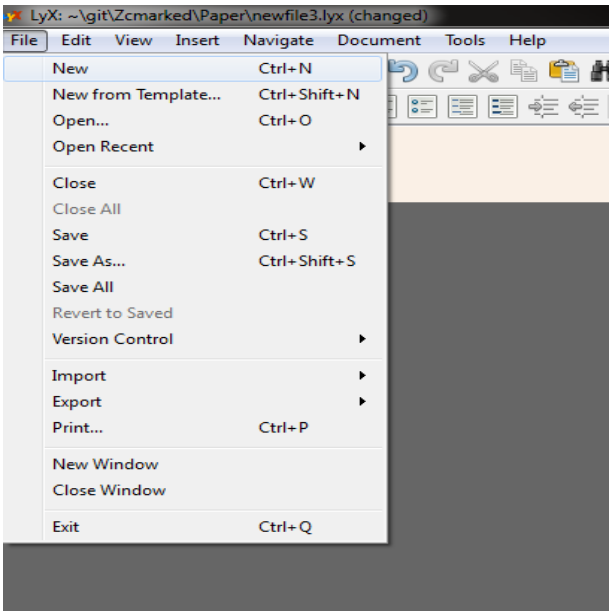


Figure 1: Creating a new empty document.

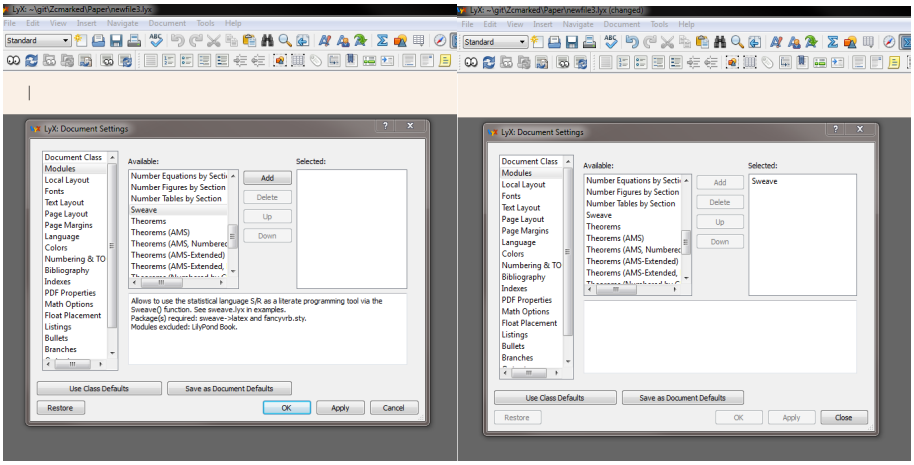


Figure 2: Adding Sweave module to the document with Document/Settings/Modules.

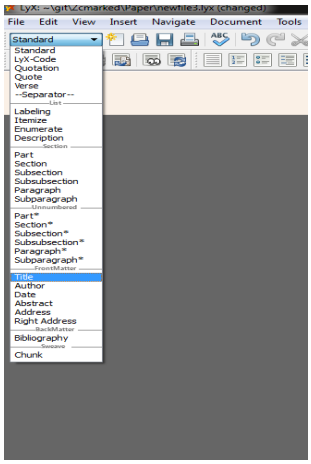


Figure 3: Adding a title to the document.

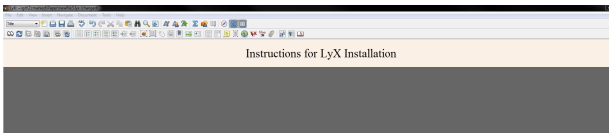


Figure 4: Entering your title.

3 Adding Title, Author, Date and Abstract

You could just begin typing text into your document but you will probably want to give it a title etc. The various structures in a LaTeX document are created with the button in the top left that will be displayed as Standard when you first open the document. These structures are called environments so I'll refer to this as the environment button. Standard represents the bulk of the document. To change the environment for a portion of the document, like a title, you open the drop down box by clicking on the button and then selecting the environment you want, a title in this case (Figure 3).

This will change to the Title format for the particular document class and you can enter the title for your document (Figure 4). Make sure to hit enter after entering the text for your title. You will notice that the environment button will return to saying Standard but if you click anywhere on the title it will change to title. You can then do the same thing with Author. A date will automatically be entered and it will always use the current date. If you want to add a fixed date then you can use the Date environment in the same way as Title and Author. If you don't want a date to appear then add **\date{}** into your LaTeX preamble (Figure 5) or under Document ▷ Document Settings ▷ Document Class check the box that says Suppress Default Date. Any other LaTeX that affects the entire document can be added to the preamble.

Following the date if any, most articles contain an abstract. The abstract is started with the Abstract choice on the environment button. It will center the word Abstract and then you can type one or more paragraphs for the abstract. You can end the abstract by pressing the enter key twice or selecting Standard on the Environment button.

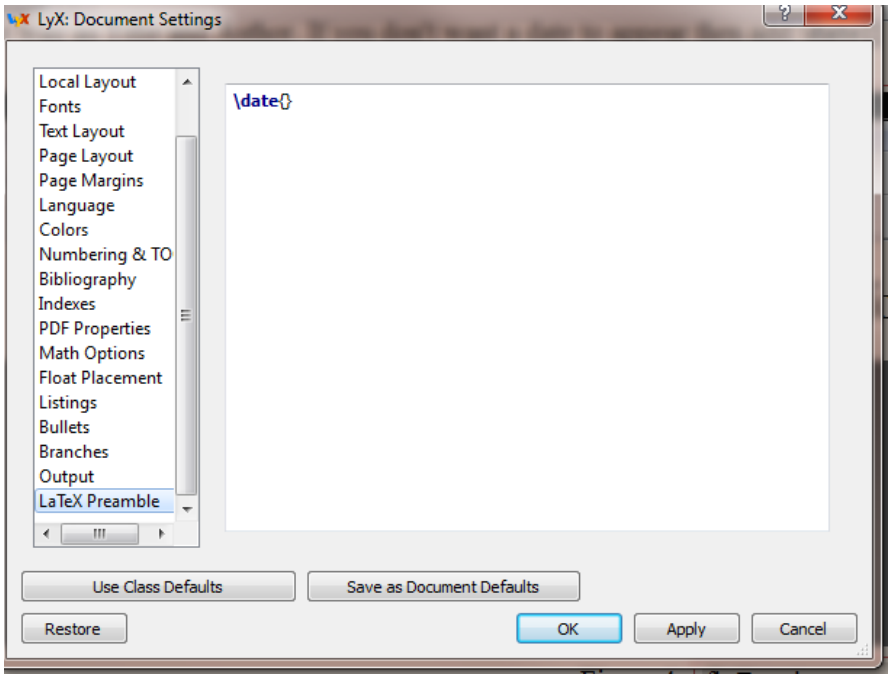


Figure 5: Adding LaTeX to the preamble.

4 Adding sections and text to your document

You can add numbered or unumbered sections to your document and within sections, sub-sections, sub-sub-sections etc. Again this is done with the environment button that was used to create title and author. Simply select Section for numbered sections and Section* for unnumbered sections (Figure 6). Then enter the text for your section header and press enter to return to standard entry. LyX/LaTeX takes care of the numbering for you.

After you have entered a section header you can enter whatever text you want. By default for English language documents, the first paragraph after a section or sub-section is not indented. If you want that first paragraph to be indented add \usepackage{indentfirst} to the LaTeX preamble under Document Settings.

Below I pasted the text from my LyX installation instructions (Figure 7) but you can also enter text manually. It was rather messy and I wanted to make it into an enumerated list so I did that by selecting the text and selecting the "enumerate" environment (Figure 8). It added numbers for each "paragraph" (separated by returns) in my text and then I cleaned up the text removing the numbers that were in the original text.

If you hit the enter key anywhere in the list it will start a new numbered entry at that point. The same thing occurs if you hit enter at the end of the list. To complete the list, simply change the environment back to Standard at the end of the list (Figure 9).

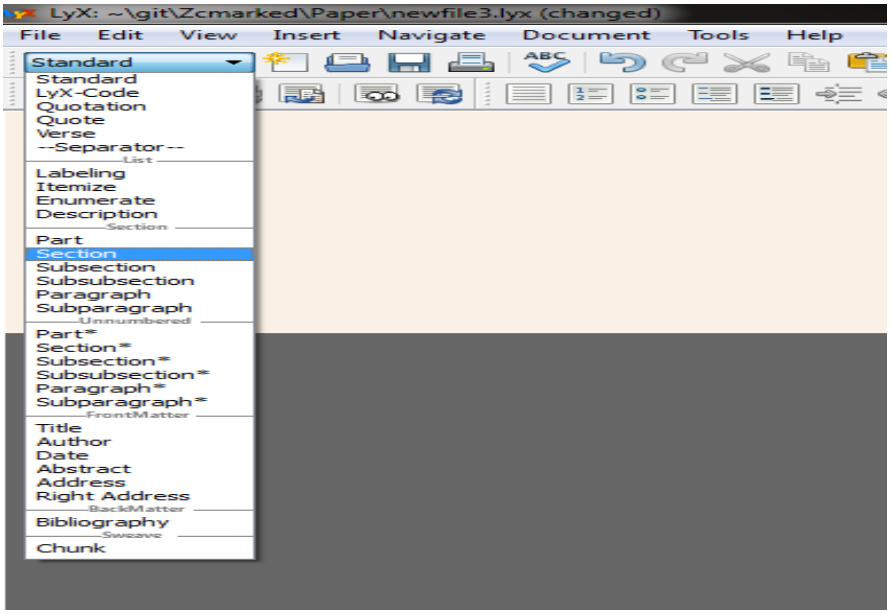


Figure 6: Adding a numbered section.

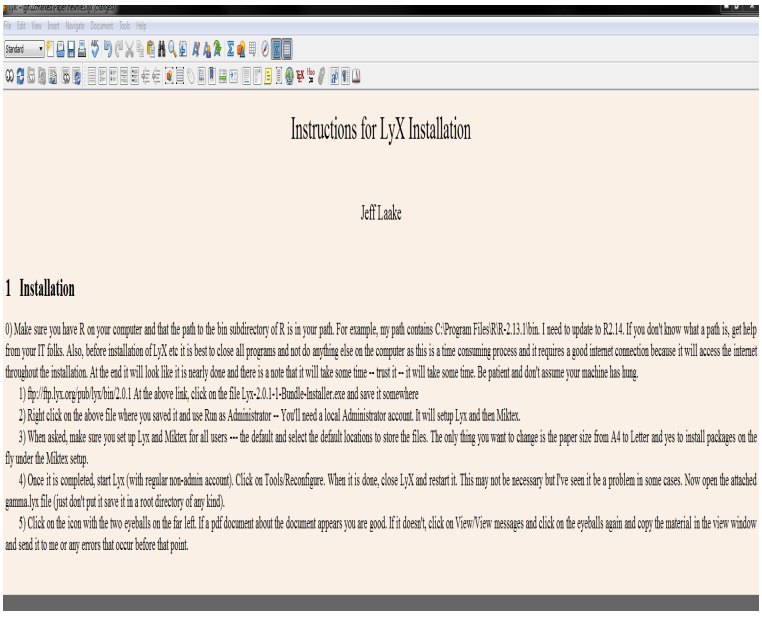


Figure 7: Adding text.

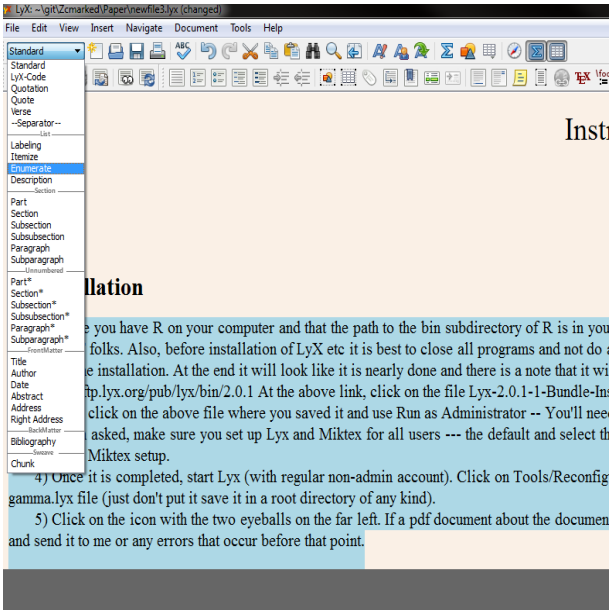


Figure 8: Enumerating a list.

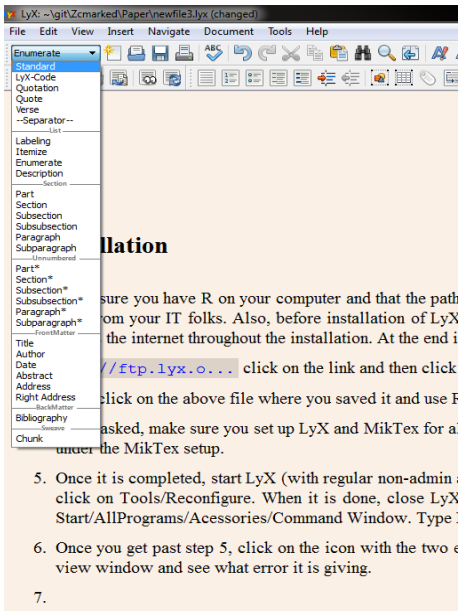


Figure 9: Ending a list.

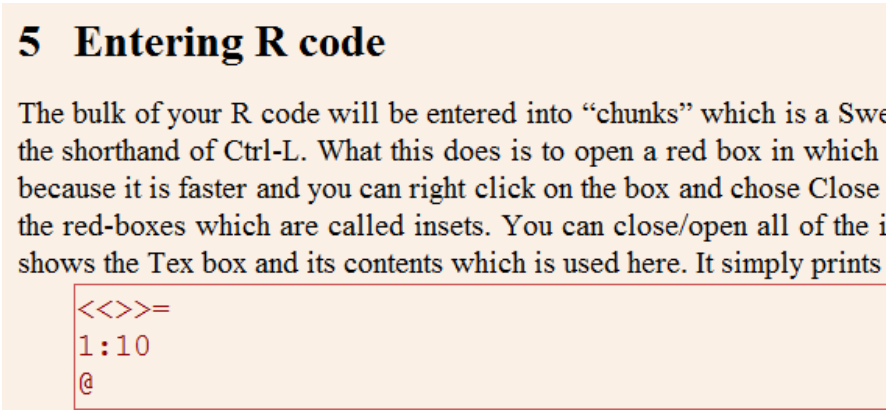


Figure 10: Entering R code.

5 Using Sweave in a document

5.1 Entering R code

The bulk of your R code will be entered into "chunks" which is a Sweave (noweb) construct. One way to create a chunk is select the chunk environment with the environment button. I prefer to use Insert ▷ Tex Code from the menu or the toolbar; although I routinely use the shorthand of Ctrl-L. What this does is to open a red outlined box in which any LaTex code can be entered including a chunk. These boxes are referred to as evil-red-text (ERT) in LyX because Tex code can be a little daunting at first. I use this approach because it is faster and you can right click on the box and chose Close Inset which collapses the box so it is not in the way when you are editing the rest of the document. Chunks created with the chunk environment are not closed and the Sweave manual suggests that the chunk environment is not always stable. Closing insets also closes figures and other objects contained in the red-boxes (insets). You can close/open all of the insets from the View menu item.

When you create a Tex code box, it is empty and you need to create a specific structure for the chunk. In its simplest form, the chunk begins with <<>>= on the first line and ends with @ on the last line. Anything on the same line beyond the @ is ignored. An optional label and options can be specified within the <<>> as shown below in section 5.3. The R code is on the lines in the middle. Each complete line of R code should end with an Enter (carriage return (CR) which is a carryover from typewriters). Typically you will hit the enter key if you enter the code with the keyboard. But when you copy R code from somewhere else, make sure to use Paste Special /Plain text (Ctrl-Shift-V) and not the regular paste (Ctrl-V). The former will add the CR and the latter will not. You will get error messages when you Sweave the code if there are no CR values between the lines.

Code that you enter into an R console can be entered into a chunk. Figure 10 shows the contents of a simple chunk in the Tex box used to get the output below. This code simply prints the sequence 1:10 (the numbers 1 to 10) and nothing is saved.

```
> 1:10

 [1]  1  2  3  4  5  6  7  8  9 10
```

Here is a more complex example in which I use the iris data for a regression and display a summary:

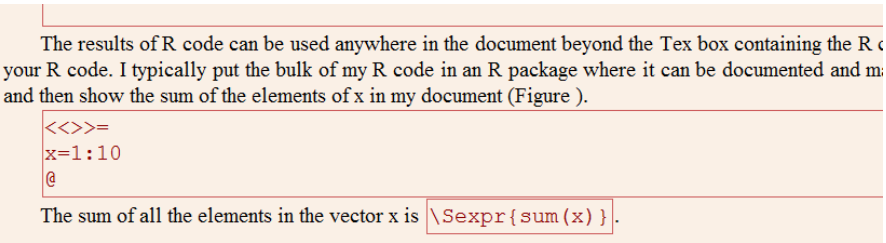


Figure 11: Using R code in an Sexpr to display results in the text.

```
Call:
lm(formula = Sepal.Length ~ Sepal.Width, data = iris)

Residuals:
    Min      1Q  Median      3Q     Max
-1.5561 -0.6333 -0.1120  0.5579  2.2226

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   6.5262     0.4789   13.63   <2e-16 ***
Sepal.Width  -0.2234     0.1551   -1.44    0.152
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.8251 on 148 degrees of freedom
Multiple R-squared: 0.01382,        Adjusted R-squared: 0.007159
F-statistic: 2.074 on 1 and 148 DF,  p-value: 0.1519
```

5.2 Using R code results

Think of the R code in your article as if it was a script being run in the R console. It is interpreted from top to bottom. The results of R code can be used anywhere in the document beyond the Tex box containing the R code. Here is an example in which I create a variable x and then show the sum of the elements of x in my document (Figure 11).

```
> x=1:10
```

Using \Sexpr{sum(x)} in a Tex box, I can create text like: The sum of all the elements in the vector x is 55. If you don't like typing \Sexpr{}, you can use the Insert ▷ Custom Insets ▷ S/R expression and type the code (e.g. sum(x)) in directly.

As with the R console, printed output is displayed with a default number of digits. Often the simplest approach is to use the round function. The value of \Sexpr{sum(x)/6} is 9.16666666666667; whereas the value of \Sexpr{round(sum(x)/6,1)} is 9.2 . The functions formatC or sprintf are better to use when you are constructing tables in which you want each value to be displayed with a constant number of digits (eg. you want to display 0.57 and 1.00 rather than 0.57 and 1).

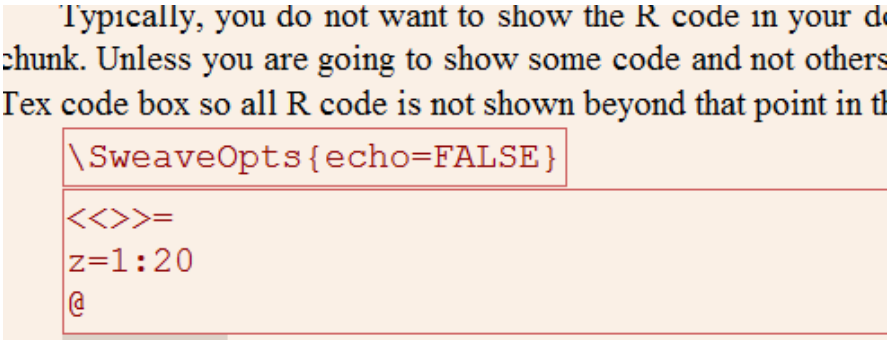


Figure 12: Turn echo off for all Tex boxes past the \SweaveOpts, such that the R code is not shown in the pdf file.

5.3 Chunk options

Typically, you do not want to show R code in your document unless you are writing a book about R or writing a tutorial like this one. You can prevent showing the code by using the option <<echo=FALSE>>= when you define the chunk. Unless you are going to show some code and not others, you probably will want to set this behavior for all your chunks. You can do that by entering SweaveOpts{}. For example, you can enter \SweaveOpts{echo=FALSE} into a Tex code box so all R code is not shown beyond that point in the output. See Figure 12 which shows the contents of this LyX file which has a chunk after the SweaveOpts is set but nothing is shown.

Even though the code is not shown, I can still use and display the results as in "the sum of the elements of z is" 210. The contents of the Tex code box for the sum of z is \Sexpr{sum(z)}.

The various options that can be used in SweaveOpts and inside the <<>> of the chunk definition are documented in the R help file for RWeaveLatex. I'll describe some of the more useful options here. Options are specified with the form key=value where key is the name of the option. Options are separated by commas. A chunk label can be entered as label=value or by simply using the label value as the first element contained in the <<>>. For example, you could use <<mychunk,echo=FALSE>> to assign the label mychunk and to prevent echoing the code to the document. If you give the chunk a label, it will appear in View Messages which is useful for tracking errors.

Setting echo=FALSE will prevent the code from being shown but will not supress any output that results from calling functions. Sometimes you can simply assign the results of the function to a object (eg. x=dev.off()) so it is not printed but in some cases that will not work. For example, some packages will print a command to the console when you load them with the library function and setting the quiet argument to FALSE will not suppress the output. If you want to suppress all output from the R code, set the option results=hide. The results option can also be set to tex (i.e. <<results=tex>>) when the output of the R function is TeX code. This will be used in section 7.5 with the xtable package for automatic table creation.

If you are echoing R code as in this document and you want your comments (#) to be echoed as well, you should set the option keep.source=TRUE for the chunk or as a global option with SweaveOpts. If you are using the TeX box soley to demonstrate how R code should appear but you don't want to evaluate the code, use the option eval=FALSE. Other options for generating figures are described later.

5.4 Sweave processing

In processing a document that contains the Sweave module, LyX creates an .rnw file that contains the R code in the rest of the document. LyX runs Rscript to sweave the document which converts the .rnw file to a .tex document. The .tex file will contain the results of running the R code. After creating the .tex, LyX calls MikTex to process the code to create a pdf or other form of document format (eg dvi). Each of the intermediate files and the final pdf can be found in the temporary directory that LyX uses for processing. The location of the temporary directory can be set with Tools ▷ Preferences ▷ Paths. You can also have these files saved to the directory containing your manuscript by using File ▷ Export ▷ Sweave to get the .rnw file and File ▷ Export ▷ Latex (pdflatex) to get the .tex file which is constructed after running Sweave on the .rnw file.

Even though intermediate files are stored in a temporary directory, the working directory for R (getwd() in R) is the directory containing the LyX document. Any files that you create with the R code will be contained in the same directory as the LyX document. If you save or load a workspace for the working directory it will also be contained in the same directory. You can hard-code alternate directories but you'll will lose that advantage that the paper can be copied into a different location/machine and is not dependent on any particular directory specification.

5.5 Strategies for using R code in Sweave documents

The simplest and safest way to use Sweave is to nest all of your R code into chunks in your LyX document. I say safest because any time you construct the document it will use the code in the document and is thus always reproducible. If you know you haven't changed the original document it should produce the same result and you can use a source versioning system (e.g., git) to track any changes in the LyX document because it is a text document. The only external changes that can affect reproducibility are changes in R or R packages that you are using and technically you can maintain both the R version and its packages to avoid any external changes. Changes in R and R packages is a hazard with any of the approaches I describe here. While including all code may be the safest approach, some documents might contain so much code that it becomes cumbersome to work with the document even though you can close the chunk insets. Also, code maintainability can become an issue if you have lots of code throughout the document. Below I discuss some strategies for dealing with code in Sweave files.

Another simple strategy is to include all of the R code in a separate R script file (e.g. mysource.R) and have a single chunk that contains source("mysource.r"). This will work fine as long as you ensure that you don't make modifications to the R script after producing the manuscript. Again source versioning software can be used to track changes and dates in the R script file as well as the LyX document. You can save the .tex file (also a text file) and compare any newly generated .tex file for differences if there is any concern about code being changed. Taking this one step further, I put the bulk of my code into functions and build the functions into an R package where it can be easily documented and maintained. In the document I include the code to do the analysis which includes function calls and code that isn't suited for a function. This is similar to the script approach except that the script is formalized into functions that can be more fully documented. Also, the packaged code can be used in a similar analysis for a different article. The danger here is the same as with CRAN packages, that you change the package and the results are no longer reproducible. With packages that you create, you have far more control over what is changed, but I have experienced a case where I had to track down changes to a package that I had made that changed the results in a manuscript I had produced previously. Again source versioning of the package code is very helpful and it is probably wise to keep a separate archived copy of your package with the manuscript once it has been completed.

Bottom line: If you have a small to moderate amount of code, put it all in the LyX document. If you find that your code is getting out of hand and you can't find what you are looking for, consider using one of the above strategies. If you want to export all of the existing R code in a LyX document into a single R script file, select (File ▷ Export ▷ R/S code).

5.6 Debugging in LyX

Inevitably you will run into situations in which you try to convert your article into a pdf and LyX responds with an error message. I have never found LyX error messages to be very informative. If you believe that the error is in your R code, then select View ▷ View Messages. A split screen will appear with a log file that will contain any error messages that occurred in Sweaving the document. Often this will be a syntax error or an unknown variable and you should be able to figure it out. If you use labels for your chunks, the label will help you locate which chunk contained the error rather than using the assigned numeric values based on their order in the manuscript. If the error is not in the R code, it may be helpful to examine the .rnw and .tex files that are in the temporary directory as set in your path specifications which can be found under Tools ▷ Preferences ▷ Paths.

In writing documents like this I discovered an odd behavior when you wrap text around a chunk. If you want the text following the chunk to be a new paragraph then you simply hit enter (return) after the chunk and all will work. To do that place the cursor outside the red box on the side and then hit enter. It can only be placed on the first line on the side. If you remove the enter such that the text following the chunk is not indented, it will look fine in your LyX file but it will not appear in your PDF and if you look at the .rnw file in your temporary path, you'll see that the line of text following the chunk is on the same line as the @ which is for a comment so the text is ignored. A similar problem will occur if you don't hit enter on the line preceding the chunk. If you don't want to indent the text following the chunk, you can use Ctrl-Enter (line-break) but a better approach is to use Enter and then to change the paragraph settings (Edit ▷ Paragraph settings) by unchecking the Indent box.

Sometimes errors occur in the R code will appear in the View Messages but the pdf is still created. I'm not entirely sure what causes that behavior. Regardless, if you make a change in your code and the expected change does not occur in your document check View Messages.

One of the best ways to avoid debugging in LyX is to place your chunks in locations where they won't cause any of the problems I described. I put the code for a section or sub-section right after the section header and before the text. You could always put all of your code in a single TeX box and the beginning of the document if you prefer. Also, I suggest running and debugging code in R where it is easier to see and correct the errors. Then once it is working copy it into the TeX box. File ▷ Export ▷ R/S code is useful for getting access to all the code and pasting or sourcing it into an R console.

6 Adding Figures

6.1 Creating a figure with R code

If you are using Sweave, then many of your figures will be plots and other figures showing the results of your calculations. The Help ▷ Specific Manuals ▷ Sweave Manual provides some information on including figures and I'll use one of their examples and then show additonal approaches.

The easiest approach to generate a figure is to use the chunk option fig=TRUE and include the R code to generate the figure. The default graphics device is pdf but the options eps, jpeg,

png and pdf can be set to TRUE/FALSE to specify the graphics driver. If you use a par function to set graphics parameters, it only holds for that graphic because a new device is opened for each figure chunk. The following example creates a pairs plot within a graphics device window that is 4.5 inches by 4.5 inches. The file name for the pdf plot will be the filename of the document followed by iris-pairs which is the chunk label name. The file that is produced for the plot will be in the LyX temporary directory. When the plot is displayed in Figure 13 the width of the displayed plot is the default of 0.8*textwidth. Figure 14 demonstrates the distortion that can occur if the width is too narrow.

```
<<iris-pairs,fig=TRUE,width=4.5,height=4.5>>=
pairs(iris, col = iris$Species)
```

You can make plots use the specified width or full text width by setting
\setkeys{Gin}{width=\maxwidth}
in a Tex box at the beginning of the document and defining maxwidth by including the following in the LaTeX preamble. See Help ▷ Specific Manuals ▷ Sweave Manual:

```
%% maxwidth is the original width if it's less than linewidth
%% otherwise use linewidth (to make sure the graphics do not exceed the margin)
\def\maxwidth{%
\ifdim\Gin@nat@width>\linewidth
\linewidth
\else
\Gin@nat@width
\fi
}
```

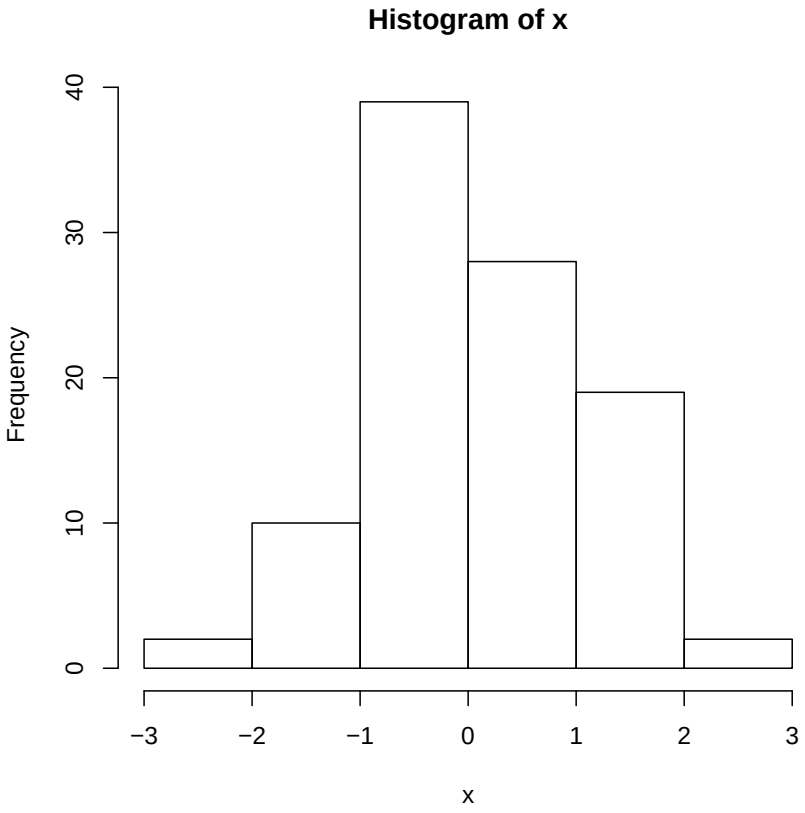
On occasion the code can become messy for producing the graphic and you may want to include it in its own chunk and then refer to the chunk label to create the figure. The following is a very simple example that shows the stucture you would use with the result shown below:

```
<<normhist>>=
x=rnorm(100)
hist(x,nclass=5)
```

```
<<plotnorm,fig=TRUE>>=
<<normhist>>
```

```
> x=rnorm(100)
> hist(x,nclass=5)

> x=rnorm(100)
> hist(x,nclass=5)
```



This is an example of reusing a code chunk and for plots it is only probably useful for writing tutorials, but can be useful more generally to modularize code. Recognize that the code is run twice so you don't want it to be a complex calculation and it is best used when the code does need to be re-run.

A better approach for figures is to specify a graphics device with functions like pdf(), eps() and then using the LaTeX command \includegraphics in the figure.

```
<<normhist>>=
x=rnorm(100)
pdf("myhist.pdf")
hist(x,nclass=5)
dd=dev.off()
```

```
\includegraphics[width=5in,height=5in]
{myhist.pdf}
```

In the chunk, a pdf device was opened and the file was named myhist.pdf. The file location will be in the directory with the LyX file and not in the temporary directory which is useful if you want each figure in a file separate from the final document. The dev.off() closes the device but since it returns a value it is assigned to the dd object so that is not printed out. I could have used results=hide to do the same thing. The second TeX box creates the figure with the LaTeX command \includegraphics. The width and height here are of the size of the plot in the document and not the graphics device and the units here are specified as inches. If I wanted to set the width and height of the graphics device I would have done so in the pdf() function call.

Notice that the filename is in quotes in the pdf() call but not in the {} for the \includegraphics. Get these wrong and you'll get some strange errors. The functional chunk is below and the resulting histogram is in Figure 15.

```
> x=rnorm(100)
> pdf("myhist.pdf")
> hist(x,nclass=5)
> dd=dev.off()
```

So far I've not discussed figure placement and I've actually used 3 different approaches in this document. The first case was the histogram which was inserted in the document with the fig=T chunk. It was placed directly after the text that it followed. That is quite simple and not useful for a polished document because a caption can't be assigned to the figure. The best approach is a Figure Float which is created with the Insert ▷ Float ▷ Figure as shown in Figure 16. Floats are explained more below. The TEX processor will place a float in the best possible place it can to be close to the reference to the figure. Sometimes you may want to force the location of a figure to a particular page. I do that by inserting a clear page before and after the figure float (Insert ▷ Formatting ▷ Clear Page). Typically that works best when all of the figures and tables are positioned at the end of a document. The following three figures have all been placed on individual pages which explains the white space at the bottom of this page. Not what you really want to have in your document necessarily because the spacing depends on the amount of material prior to this point.

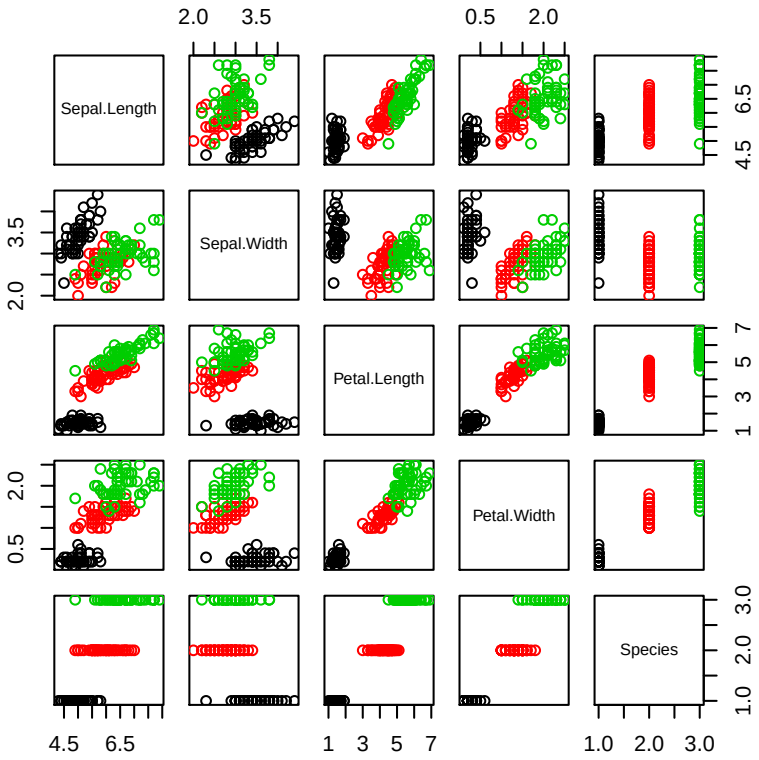


Figure 13: Scatter plot matrix of the `iris` data with width=4.5.

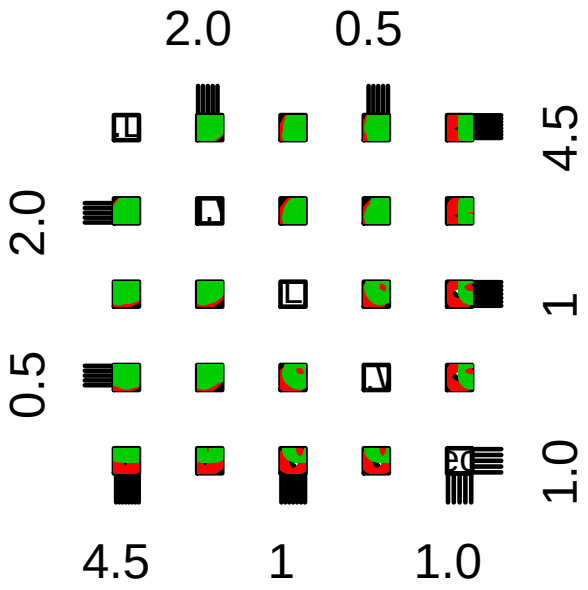


Figure 14: Scatter plot matrix of the `iris` data with width and height =2.

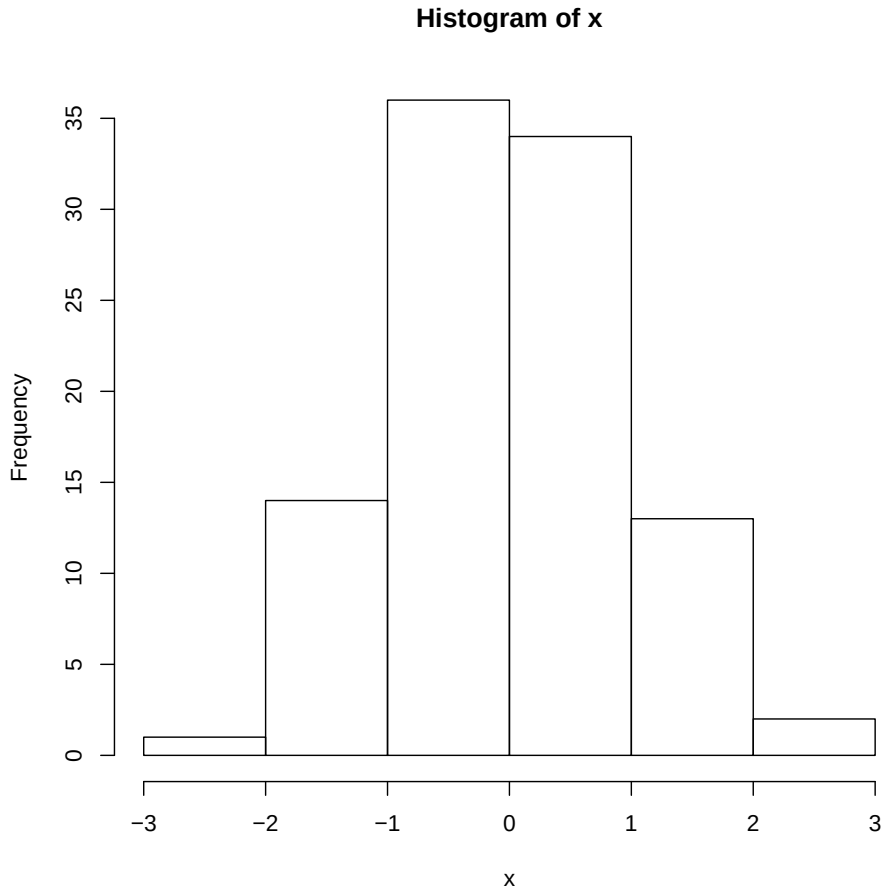


Figure 15: Histogram of normal variable created with pdf() and \includegraphics.

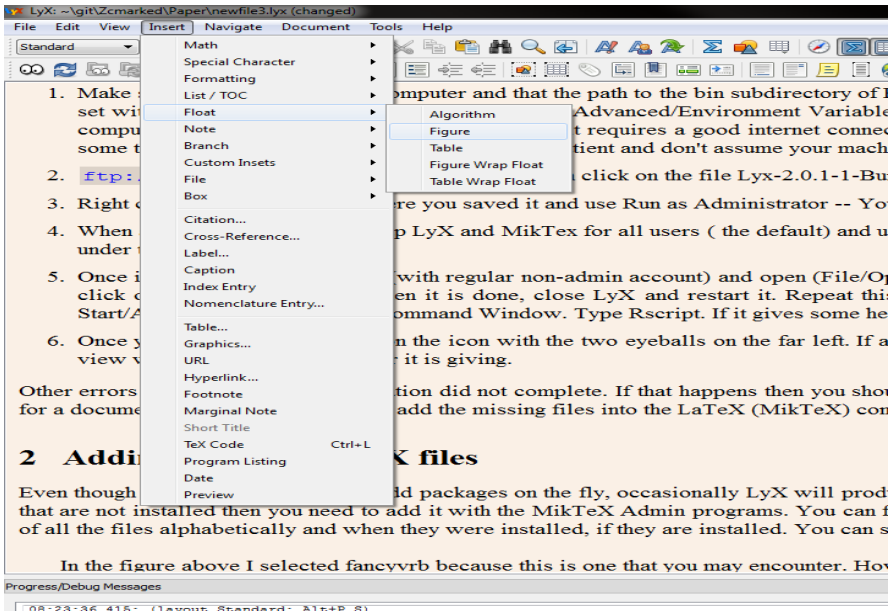


Figure 16: Adding a floating figure.

6.2 Creating a figure with external file

Sometimes you'll want to include a map, photo or other graphic that is not created by R. Here I will show you to how to create a figure with a graphic or picture. Also, I'll show how to add a title and label to the figure and use the label to cross-reference in your document. In Figures 16 - 19 it shows how to Insert ▷ Float ▷ Figure, add a title, enter the graphic to be contained in the figure and add a label for cross-referencing.

Once you have created the figure and given it a label, you can refer to the figure in the text by inserting a cross-reference which is a pointer to the label (Figure 20). What is substituted is the figure number so you will also want to add the word Figure or Fig. or whatever should accompany the number. If the figures are re-arranged then the numbers shift in the document automatically.

7 Adding a table

A table can easily be added with tools that are similar to what you would find in any word processor. If you select Insert ▷ Table from the menu, a box will appear asking for the number of rows and columns that should be included in the table. There are many tools available for manipulating the table which will appear at the bottom of the screen if you click on the table icon at the top near the Sigma (summation sign) which displays the toolbar for equations. Once you create a table and the cursor is inside that environment, the table icons will work.

7.1 Filling a table with computed results

You can certainly fill a table with any hard-coded information that you want, but my focus here is creating tables that are filled in with the results from the R code. This isn't really any different than what was shown in section 5.2. You simply create a table with the table icon

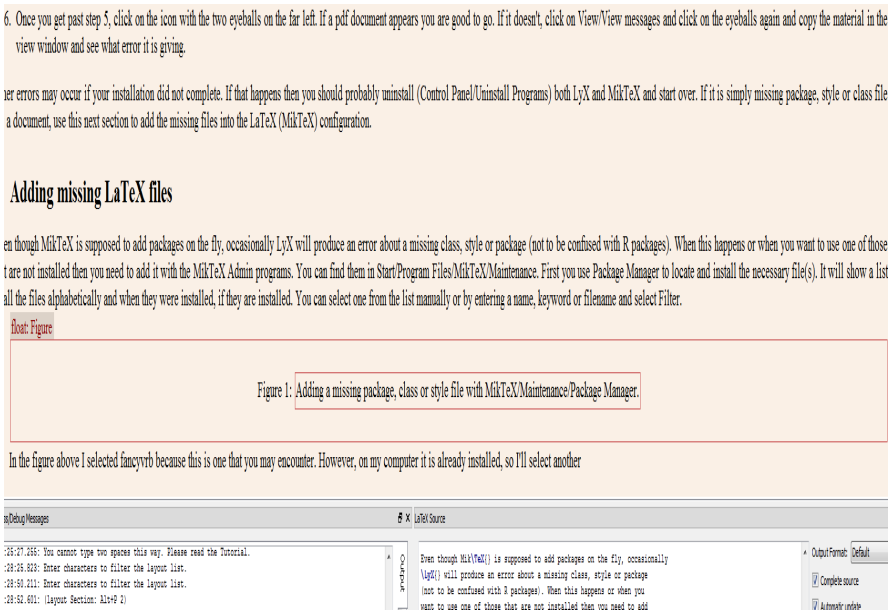


Figure 17: Adding the caption for the figure.

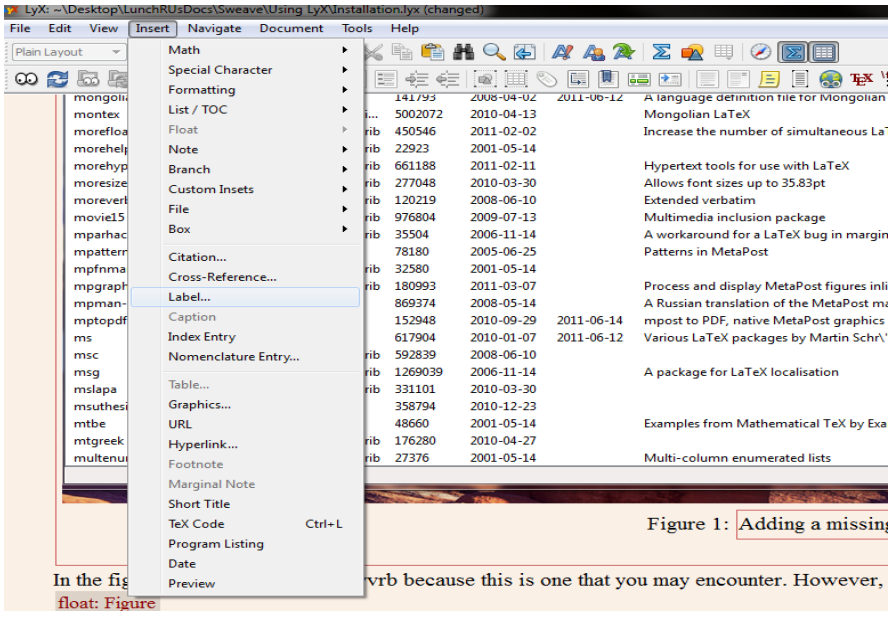


Figure 18: Adding a label to the figure for cross-referencing in the text.

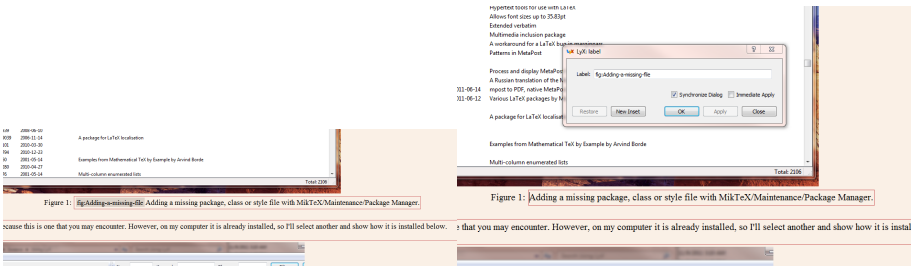


Figure 19: Entering the value for the label. Typically I use the default value that it offers which is the beginning portion of the caption as shown in the bottom portion of the figure. The result is shown in the top portion of the figure.

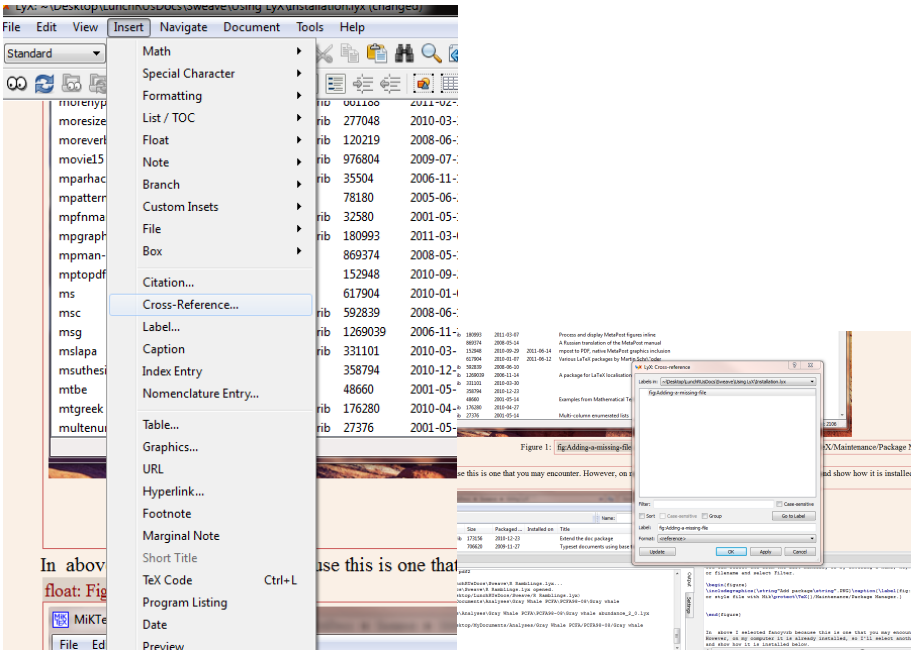


Figure 20: Using a label cross-reference in the document.

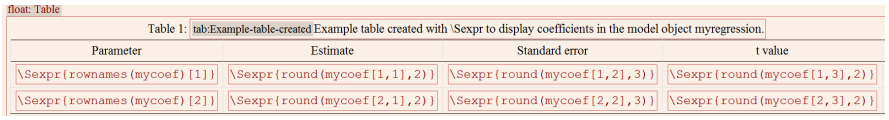


Figure 21: Contents of a table created with results of R code using \Sexpr{}.

Table 1: Example table created with \Sexpr to display coefficients in the model object myregression.

Parameter	Estimate	Standard error	t value	Pr(>|t|)
(Intercept)	6.53	0.479	13.63	6.4697e-28
Sepal.Width	-0.22	0.155	-1.44	1.5190e-01

typically within a Table Float (Insert ▷ Float ▷ Table) and fill in the table cells with \Sexpr{} Tex boxes or Insert ▷ Custom Insets ▷ S/R expression. The following example in Table 1 was created with the table icon and completed with the results from myregression in a previous example. A partial display of the contents is shown in Figure 21.

7.2 Sideways tables

On occasion tables are too wide (not so in this example) and you'll want to display them in landscape mode. Doing so is trivial. Insert a table float with Insert ▷ Float ▷ Table. Add your caption(title) and your table contents. Then right-click on the float:Table and select Settings from the list. Click on Rotate Sideways and Apply as shown in Figure 22 and (sideways) will be added next to float: Table. To get the table centered on the page, I added a Tex box containing \centering in the Table Float. The result is shown in Table 2.

7.3 Long table

Sometimes tables are also too long to fit on a single page and need to span pages. This is a little more difficult but with the instructions and template I provide, it should be quite doable by following these steps:

1. First create a Table with the Table Icon but **not** with Insert ▷ Float ▷ Table. Long tables will not work in a float because they span pages.

2. Add 3 extra rows to the header portion of the table.

3. Add the text for your column labels in the second and fourth rows. The values in the fourth row will be repeated on the spanned pages.

4. In the third row and first column add a Tex box (Ctrl-L) and type in \caption{continued}\\%. What is in the braces{} is up to you because this text will be added to the spanned pages with Table and its number. If you don't want the latter use \caption*{continued}\\%.

5. Next put the cursor in the first row of the table and right click with the mouse and select More ▷ Settings and then the tab for Longtable. Check the Use long Table box and the Caption on box.

Table 2: A sideways table example with a manually created table.

Parameter	Estimate	Standard error	t value	Pr(>|t|)
(Intercept)	6.53	0.479	13.63	6.4697e-28
Sepal.Width	-0.22	0.155	-1.44	1.5190e-01

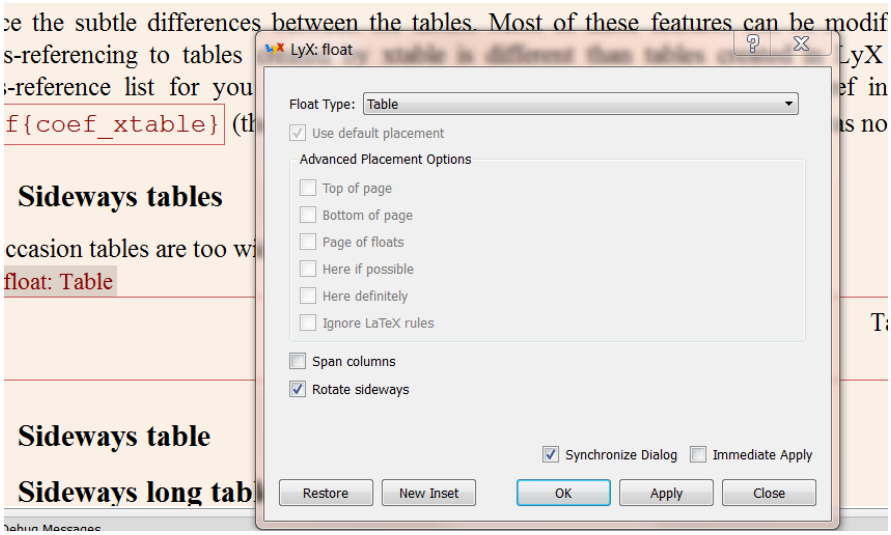


Figure 22: Making a Float Table turn sideways in landscape mode.

6. Press Apply and the table will be centered on the page and a Table #: will be followed with a red box. You can also use right or left placement instead of center by checking the box under settings and then Apply. Add your table caption in the red box.

7. Next click in row 2, check the First Header on box and hit Apply. Then click in row 3, the continuation label, check the Header on box and Apply. Repeat the same operation for row 4, the second set of column headers.

8. Change the borders to the way that you like them. Typically I highlight the entire table and remove all borders and then add the lines I want. You'll likely want a top and bottom line on row 2, and a bottom line on rows 3 and 4.

9. Now add a row at the bottom of the table and give it a top border. Click in that row and check the Footer on box and press Apply. This will add an underline on the bottom of each table as it spans pages. Don't put a lower border on the last row of the table containing data because the footer will do that and it will look darker if you apply both. You also have the option of using last footer for something specific at the end of the table.

10. Remember always to press Apply. If you move the cursor out of the row before hitting apply the action will not taken. Now to check if you have it right, move the cursor in the different rows and in the Settings you'll see the checked boxes change.

11. Click Close and view the document to see the Table. If something is not being applied correctly, go back into settings and check the values of the various rows. You may have forgotten to press Apply.

Now if you read all of those steps and it seems entirely too complicated, there is a simpler solution. Copy the longtable example from this LyX document and paste it into yours. Change titles, column headers etc. Add or delete columns and rows and enter the information and you should be good to go.

You can set the caption width of floats by using \setlength{\LTcapwidth}{1.5in} in a TeX box where 1.5in is an example width used in Table 3. Once it is set it will continue for all ensuing

tables until it is reset. You can reset it to the text width with \setlength{\LTcapwidth}{\textwidth} in a TEX box.

Table 3: A long table example with a very long caption. If you want the caption to stay within the boundaries of the table, add a TEX box with the following contents \setlength{\LTcapwidth}{xxin} where xx is the width in inches. This example used 1.5 inches. This setting will be maintained from this point throughout the document until it is reset except for xtables.

Field1	Field2
1	a
2	b
3	c
4	d
5	e
6	f
7	g
8	h
9	i
10	j
11	k
12	l
13	m
14	n
15	o
16	p
17	q
18	r
19	s
20	t
21	u
22	v
23	w
24	x
25	y
26	z
27	a
28	b

Table 3: continued

Field 1	Field2
29	c
30	d
31	e
32	f
33	g
34	h
35	i
36	j
37	k
38	l
39	m
40	n
41	o
42	p
43	q
44	r
45	s

7.4 Sideways long table

And finally sometimes tables are too wide and too long, so you want to create a landscape table that spans pages. There is no sideways longtable, so what you have to do is use the landscape package and create a long table. Add \usepackage{pdflscape} into your LaTeX preamble with Document ▷ Settings ▷ LaTeX Preamble. If you don't have that package, you'll need to add it with MikTeX. Create your long table as shown above and then add a TeX box containing \begin{landscape} prior to the table and another containing \end{landscape} after the table. An example is shown in Table 4. A new page will be started when it encounters the landscape mode, so you will have to do some thinking about where to locate the table if it is in the middle of the document.

Table 4: A sideways long table example with a very long caption with its length reset with the setting in the prior table.

Field1	Field2
1	a
2	b
3	c
4	d
5	e
6	f
7	g
8	h
9	i
10	j
11	k
12	l
13	m
14	n
15	o
16	p
17	q
18	r
19	s
20	t
21	u
22	v
23	w
24	x
25	y
26	z

Table 4: continued

Field 1	Field2
27	a
28	b
29	c
30	d
31	e
32	f
33	g
34	h
35	i
36	j
37	k
38	l
39	m
40	n
41	o
42	p
43	q
44	r
45	s

7.5 Using xtable

In many cases it is simpler and better to use the xtable package in R to create the table. It is better in the sense that the size of the table can change automatically as needed. For example if another year of data were added for an analysis, the table created manually would need to be manipulated whereas the one created by xtable would not. Below is an example in which same coeffcient table for myregression is displayed with xtable. The chunk contains the option results=tex, so the result of the R code is subsequently processed as Tex.

```
> library(xtable)
> print(xtable(mycoef,caption="Coefficient table displayed with xtable.
+ I have added this text to see what happens \nwith a very long caption on an xtable."
+ ,label="coef_xtable"),caption.placement="top")
```

Table 5: Coefficient table displayed with xtable. I have added this text to see what happens with a very long caption on an xtable.

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	6.53	0.48	13.63	0.00
Sepal.Width	-0.22	0.16	-1.44	0.15

Notice the subtle differences between Table 1 and Table 5. Most of these features can be modified using the many different options for xtable and print.xtable. See the R help files for xtable and print.xtable.

Fairly complex tables can be constructed using character matrices or even dataframes by knowing a few R commands and understanding that NA values in dataframes are not printed by default and likewise empty strings in character matrices. Here is one example using a character matrix which gives a table with means and standard deviations in parenthesis.

```
> means=sapply(iris[,1:4],function(x) tapply(x,iris$Species,mean))
> stddev=sapply(iris[,1:4],function(x) sqrt(tapply(x,iris$Species,var)))
> iristable=paste(round(means,2),paste("(",round(stddev,2),")",sep=""))
> attributes(iristable)=attributes(means)
> print(xtable(iristable,caption="Means and standard deviations in
+ parens of iris data classified by species"),caption.placement="top")
```

Table 6: Means and standard deviations in parens of iris data classified by species

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
setosa	5.01 (0.35)	3.43 (0.38)	1.46 (0.17)	0.25 (0.11)
versicolor	5.94 (0.52)	2.77 (0.31)	4.26 (0.47)	1.33 (0.2)
virginica	6.59 (0.64)	2.97 (0.32)	5.55 (0.55)	2.03 (0.27)

```
>
```

Cross-referencing to tables created by xtable is different than tables created in LyX because the label assigned in the xtable function (e.g., coef_xtable) will not appear in the cross-reference

list for you to select. Instead you use the LaTex command \ref in a Tex box. In this case it is \ref{coef_xtable} to construct the cross-reference to Table 5 (the value here was the result of the cross-reference and was not hard-coded). When you mix xtable tables and regular tables in LyX, the table numbers shown in LyX tables may not be correct because it doesn't know about the table numbers for xtable; however, when the tables are displayed in the final document the numbers will all be correct.

I have not worked out how to limit the length of the caption using the current version of xtable. I have modified the print.xtable function to include an additional argument caption.width which will let it be set. I have contacted the package maintainer to see if he'll include my changes in the package.

Both sideways and long tables can be constructed with xtable. A sideways xtable can be constructed setting the floating.environment argument of print.xtable to "sidewaystable". To use a sideways xtable, you must add either \usepackage{rotating} or \usepackage{rotfloat}. I didn't need to include it in this file manually because \usepackage{rotfloat} is added automatically when I used the sideways option of a regular table that LyX created. However, LyX does not know that you are creating a table with xtable, so unless you also have a sideways table constructed with LyX, you have to include the usepackage manually into the preamble. The regression coefficient example as a sideways xtable is shown in Table 7.

```
> print(xtable(mycoef,caption="Coefficient table displayed sideways with xtable."
+ ,label="coef_xtable_sideways"),caption.placement="top",
+ floating.environment="sidewaystable")
```

A long table is done in a slightly different manner and most importantly they cannot be floats so you'll have to be careful about their placement. An example using a dummy dataframe can be done as shown below with the output in Table 8. Notice that the table numbers are in the correct order for the sideways and long tables constructed with xtable, but because the long table doesn't float and the sideways table does not, the long table is shown before the sideways table, so they are out of order numerically. This is not a problem if you put all of your tables at the end of the paper as you do with most manuscripts because they can be put in the correct order. But if you are going to spread them throughout the document, you'll have to be cognizant of placement and may need to use some clever placement of page formatting. When I went to write this section I had no idea how to get xtable to repeat the column headers on the following page as in section 7.3. From reading the documentation of print.xtable I knew that I needed to use the add.to.row argument to add TeX code but I didn't know what to add because my TeX knowledge is still rather limited. Your first response should be "search the internet" because it contains a wealth of information on TeX. In my internet browser I searched for "xtable longtable repeat header" and on the first link I found the solution shown below. It isn't exactly the same format as described in section 7.3 because it uses a footnote that the table is continued and repeats the Table header on the following page but I'm sure with a little more looking and dinking with TeX you could derive the same results. This is where it is handy to examine the TeX file in your temporary directory that LyX produces .

```
> x=rnorm(50,0,1)
> y=rnorm(50,10,25)
> addtorow          <- list()
> addtorow$pos      <- list()
> addtorow$pos[[1]] <- c(0)
> addtorow$command  <- c(paste("\\hline \n",
+                "\\endhead \n",
```

```
+                 "\\hline \n",
+                 "{\\footnotesize Continued on next page} \n",
+                 "\\endfoot \n",
+                 "\\endlastfoot \n",sep=""))
> print(xtable(data.frame(x=x,y=y),
+ caption="Dummy dataframe displayed as a longtable with xtable."
+ ,label="coef_xtable_long"),caption.placement="top",
+ include.rownames=FALSE,add.to.row=addtorow,
+ floating=FALSE,tabular.environment="longtable",hline.after=c(-1))
```

Table 8: Dummy dataframe displayed as a longtable with xtable.

	x	y
	-0.02	5.91
	-0.29	29.57
	0.77	24.24
	1.16	13.84
	-1.37	4.82
	-0.31	-18.76
	1.21	18.24
	0.92	23.38
	0.67	18.36
	0.22	-10.75
	0.01	-16.06
	2.11	-27.54
	-0.73	55.91
	-0.32	3.87
	-0.09	-9.98
	0.17	12.98
	1.12	-4.22
	-0.33	2.42
	-1.11	-11.63
	-1.91	-9.49
	-0.81	46.72
	1.30	45.83
	1.16	-55.84
	0.71	-13.99
	-0.53	45.26
	-1.28	-8.93
	1.57	4.02
	-0.93	1.59
	-0.39	47.47
	-0.56	-8.74
	-1.82	-15.39
	-0.26	1.67
	1.53	10.08
	-1.15	-19.10

Continued on next page

Table 8: Dummy dataframe displayed as a longtable with xtable.

	x	y
	0.10	0.96
	0.43	3.65
	0.23	19.87
	1.90	-11.36
	1.07	29.53
	-0.70	14.97
	-1.06	24.49
	-0.07	46.66
	0.21	21.69
	-0.10	39.85
	-0.44	9.00
	-0.33	-13.39
	1.12	-9.64
	-0.67	-14.02
	-0.76	31.50
	1.00	16.90

Finally, a sideways longtable (Table 9) can be generated as in section 7.4 by placing TEX boxes containing \begin{landscape} and \end{landscape} around the code using xtable that creates the long table.

Table 7: Coefficient table displayed sideways with xtable.

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	6.53	0.48	13.63	0.00
Sepal.Width	-0.22	0.16	-1.44	0.15

Table 9: Dummy dataframe displayed as a longtable with xtable.

	x	y
	-0.06	30.21
	-0.44	36.53
	-0.20	-23.08
	-0.45	-1.96
	0.92	51.75
	-1.78	-18.16
	0.13	23.16
	0.01	-6.27
	0.05	-20.49
	0.87	17.62
	-2.08	3.41
	1.85	-21.05
	0.65	-38.54
	1.37	45.36
	-0.54	-25.04
	-0.69	-11.45
	-1.14	23.07
	0.38	3.34
	0.42	7.46
	-1.61	-18.33
	0.27	0.41
	0.58	17.01
	0.57	42.26
	-0.43	10.10
	0.15	-4.46
	-1.58	-36.12
	-0.71	49.09
	1.27	24.76

Continued on next page

Table 9: Dummy dataframe displayed as a longtable with xtable.

	x	y
	1.03	-15.24
	-0.64	19.33
	-2.05	6.85
	-0.90	-12.81
	2.11	-24.57
	-1.33	13.62
	1.61	73.06
	1.58	-6.52
	-0.12	11.68
	1.84	34.82
	-0.51	23.00
	-0.41	25.78
	-0.18	-48.16
	-1.70	50.72
	-0.04	44.01
	0.51	35.40
	-1.40	17.52
	-0.43	4.96
	0.95	-4.94
	2.41	23.24
	0.21	19.03
	1.09	57.29

7.6 List of tables

A list of tables can be automatically added to a document using Insert ▷ List/TOC ▷ List of Tables. This will work fine unless you are using long tables because the list of tables will contain an entry for each page containing a longtable which is an odd behavior. You can create your own list of tables by using the various features of cross-referencing which allow a number, text and page to be entered for the cross-reference. Below is a short example showing a manual list of tables including 2 of the tables in this document.

List of tables

8 Equations

Adding equations is quite simple with the icons provided to create various mathematical symbols. You'll use 2 types of equations: an inline equation that is part of the text and a display equation which is on a separate line(s). They are created with Insert ▷ Math ▷ Inline Formula and Insert ▷ Math ▷ Display Formula respectively. After inserting an equation the equation toolbar will appear any time the cursor is inside the blue equation box. Here is an example of an inline equation $\sqrt{2}$. And here is an example of a Display Formula:

$$\mathcal{L}_y = \prod_{i=1}^{n} \frac{\int_{c_{j(i)-1}}^{c_{j(i)}} p.(y)\pi(y)\,dy}{\int_0^W p.(y)\pi(y)\,dy}$$

If you want an numbered formula use Insert ▷ Math ▷ Numbered Formula or for an existing display formula right clicking on it will provide the option to number it. You can also add a label for cross-refrencing. Far more complex equations and sets of equations can be created with some of the other Insert ▷ Math options.

$$\varphi = \frac{\psi_{AB}}{\psi_A \psi_B} \tag{1}$$

The subscript and superscript operations can be done quickly by typing _ and ^. Use those operations for typical subscripts and superscripts and to include limits on summations, products and integrals as shown for a summation below.

$$\sum_{i=1}^{n}$$

Note that indexing for summation and integral limits will be to the side with an inline equation: $\sum_{i=1}^{n}$. The one difficulty you are likely to encounter is in the positioning of the cursor. You can use the arrow keys to move to different places in the equation or reposition with the mouse. It may take awhile to get the hang of it but if something doesn't look correct, hit the undo (Ctrl-Z), move the cursor in the equation and try again. To see the manual for equations use Help ▷ Math.

9 Adding Citations and References

Just like cross-referencing tables and figures, you'll want to include citations in the text to references in your Literature Cited/Bibliography. TEX and thus LYX works with the BibTEX

format for references. You can enter references in directly with the Bibliography environment but most likely you'll want to use bibliographic software such as EndNote or free software such as JabRef(jabref.sourceforge.net) or Mendely(www.mendeley.com). JabRef uses the BibTex format and Mendeley can export to the BibTEX format. EndNote as a BibTex output style but I've found that the file doesn't work directly so I import the BibTEX text file from EndNote into JabRef to create the BibTEX library. The process has the following steps:

1. To export from EndNote you'll need the BibTex output style, so look to see if you can find it in Edit ▷ Output Styles. If it is listed there then you are fine. If it is not select Edit ▷ Output Styles ▷ Open Style Manager and see if it is listed there but not selected. If it is check the box and then it will appear in the list. If it is not there, then you'll have to search and download the style from the ISI EndNote site.

2. Once you have the style, select the references you want to export and click on File ▷ Export and select the name of the txt file and select the output style as BibTex Export as shown in Figure 23.

3. Next open JabRef and select File ▷ Import into new database add select the .txt file in which you saved the references and click on open (Figure 24). You may get some error messages about invalid BibTexKeys and others but it should create default values for you. You may have to change some keys or you can do an autogeneration of keys under Tools in the JabRef menu.

Once you have the bib file (named graywhale.bib for this document) from any of the reference managers then you need to take the following steps:

1. First you need to attach it to your document in the location where you want your Bibliography or Literature Cited. This is done by creating a Bibliography with the Insert ▷ List/TOC ▷ BibTEX Bibliography. You need to browse for your .bib file which I keep in the directory with the paper (Figure 25). Then Rescan and Add. If it at any time you adjust the contents of the .bib file, use rescan to refresh the references available to the paper. You have the option of either showing cited references (the default) or all references in the bibliography.

2. You'll also want to select a bibliography style. With a little searching on the web you should be able to find the bib style file (.bst) for your journal if it is not already included in the list. Here I've used jwm.bst for the Journal of Wildlife Management style.

3. Also under Document ▷ Settings ▷ Bibliography you'll want to select a more typical citation style of natbib using author name and year as showin in Figure 26. The default setting is numeric indexing.

Adding citations in the document is as easy as selecting Insert ▷ Citation and selecting the citation(s) that you want to include and the format for the citation in the text (Figure). Here are a couple of examples: Pradel et al. (1997) or Pradel et al. 1997 or (Schwarz and Arnason, 1996).

The title for list of bibliographic references in the paper is References by default for an article class. You can change it with a TEX box containing \renewcommand{\refname}{Literature Cited} which changes it to Literature Cited in this example.

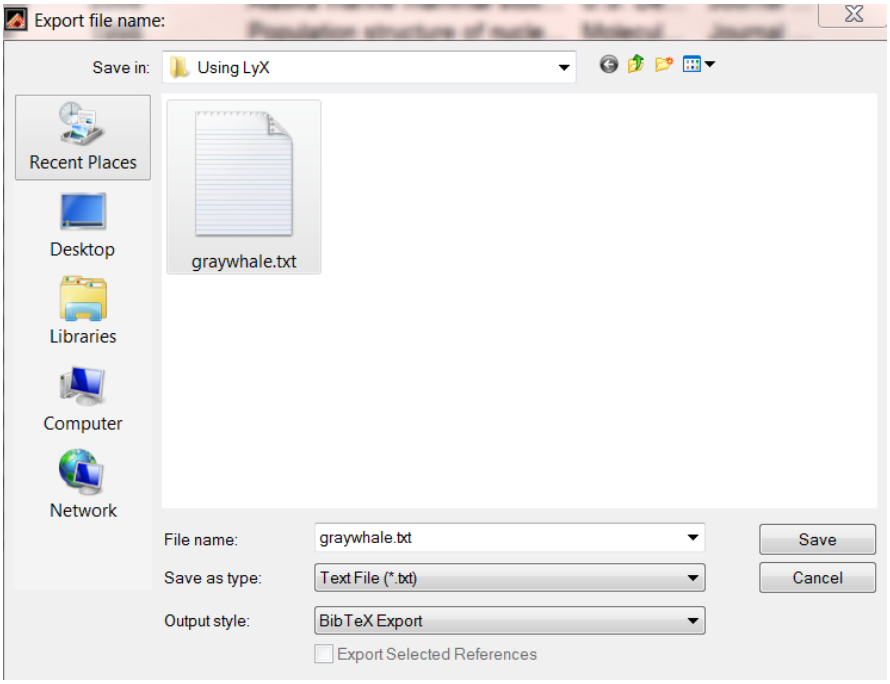


Figure 23: Example of exporting from EndNote to BibTEX format.

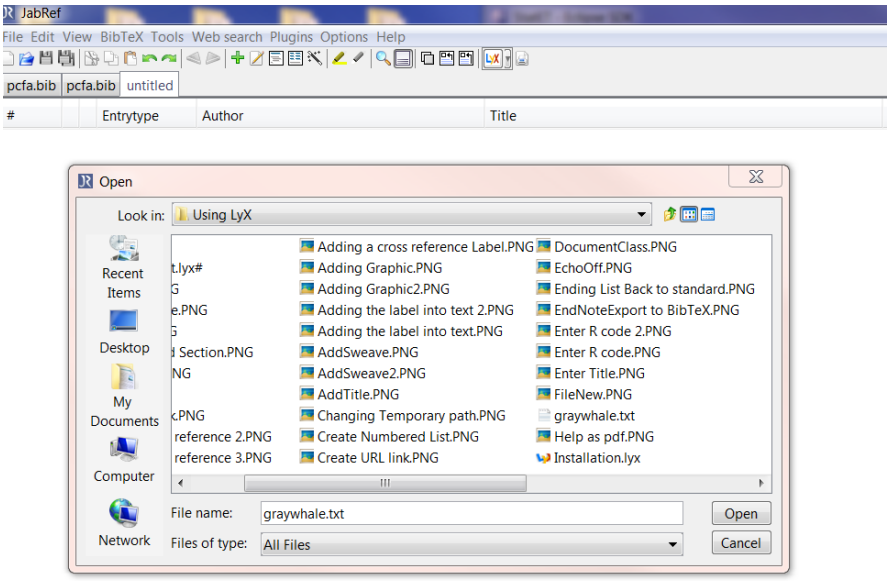


Figure 24: Example of importing into JabRef.

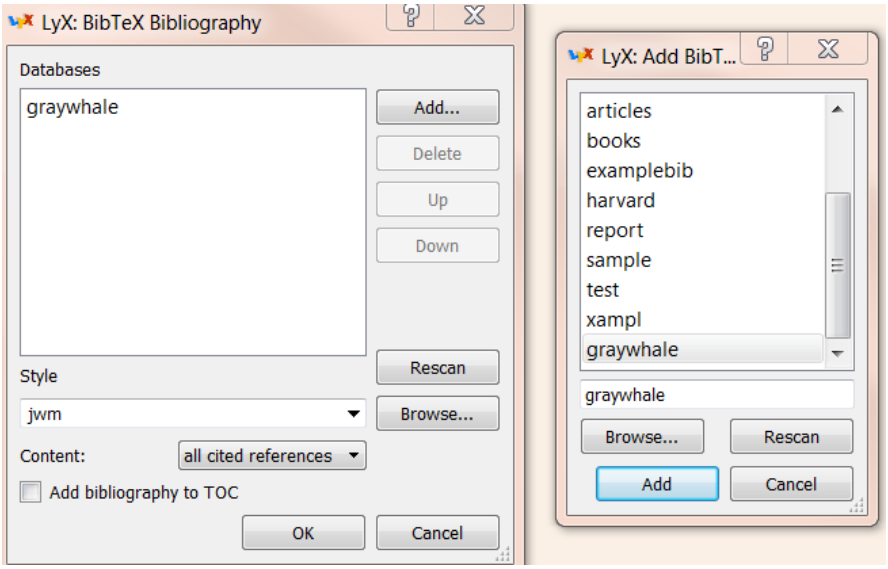


Figure 25: Adding a bibliography.

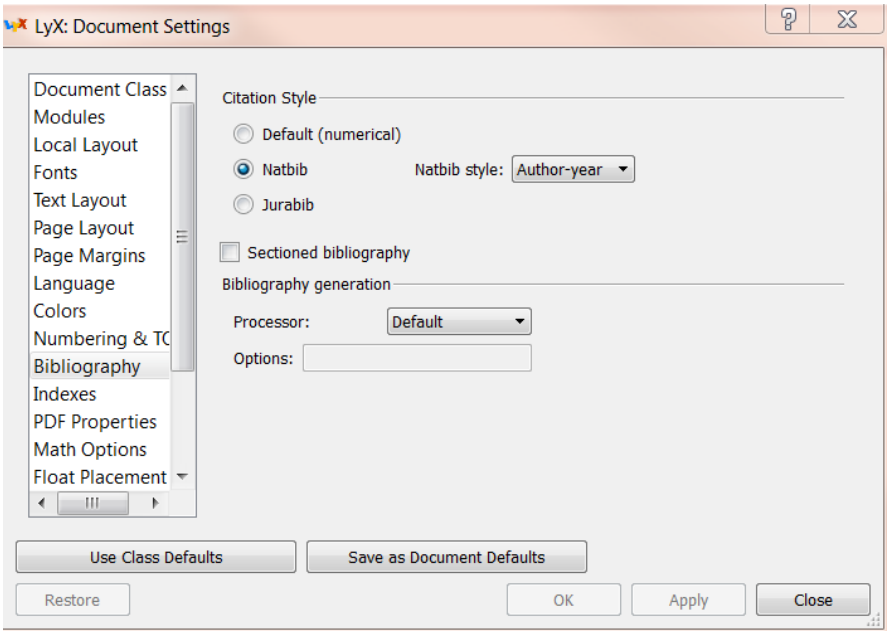


Figure 26: Selecting a citation style under Document ▷ Settings ▷ Bibliography .

Literature Cited

Pradel, R., J. E. Hines, J. D. Lebreton, and J. D. Nichols. 1997. Capture-recapture survival models taking account of transients. Biometrics 53:60–72.

Schwarz, C. J., and A. N. Arnason. 1996. A general methodology for the analysis of capture-recapture experiments in open populations. Biometrics 52:860–873.