

GS01 0163

Analysis of Microarray Data

Keith Baggerly and Bradley Broom
Department of Bioinformatics and Computational Biology
UT M. D. Anderson Cancer Center

kabagg@mdanderson.org

bmbroom@mdanderson.org

17 September 2009

Lecture 6: Sweave, R Revisited, and Affymetrix Arrays

- The Reproducibility Problem
- Installing T_EX
- Beyond Matrices
- Reading Data Into R
- Obtaining extra R packages
- Bioconductor Packages

The Reproducibility Problem

1. Researcher contacts analyst: “I just read this interesting paper. Can you perform the same analysis on my data?”
2. Analyst reads paper. Finds algorithms described by biologists in English sentences that occupy minimal amount of space in the methods section.
3. Analyst gets public data from the paper. Takes wild guesses at actual algorithms and parameters. Is unable to reproduce reported results.
4. Analyst considers switching to career like bicycle repair, where reproducibility is less of an issue.

Alternate Forms of the Same Problem

1. Remember that microarray analysis you did six months ago? We ran a few more arrays. Can you add them to the project and repeat the same analysis?
2. The statistical analyst who looked at the data I generated previously is no longer available. Can you get someone else to analyze my new data set using the same methods (and thus producing a report I can expect to understand)?
3. Please write/edit the methods sections for the abstract/paper/grant proposal I am submitting based on the analysis you did several months ago.

The Code/Documentation Mismatch

Most of our analyses are performed using R. We can usually find an R workspace in a directory containing the raw data, the report, and one or more R scripts.

There is no guarantee that the objects in the R workspace were actually produced by those R scripts. Nor that the report matches the code. Nor the R objects.

Because R is interactive, unknown commands could have been typed at the command line, or the commands in the script could have been cut-n-pasted in a different order.

This problem is even worse if the software used for the analysis has a fancy modern GUI. It is impossible to document how you used the GUI in such a way that someone else could produce the exact same results—on the same data—six months later.

The Solution: Sweave

Literate programming is an approach that embeds small program fragments within an otherwise high-quality document.

Sweave is a literate programming framework for R.

This talk was prepared using Sweave. So was [this standard report](#).

$$\text{Sweave} = \text{R} + \text{\LaTeX}.$$

Once you know both R and \LaTeX , then the thirty-second version of this talk takes only two slides.

First, we take a few moments to learn \LaTeX . (You already know R.)

L^AT_EX Document Preparation System

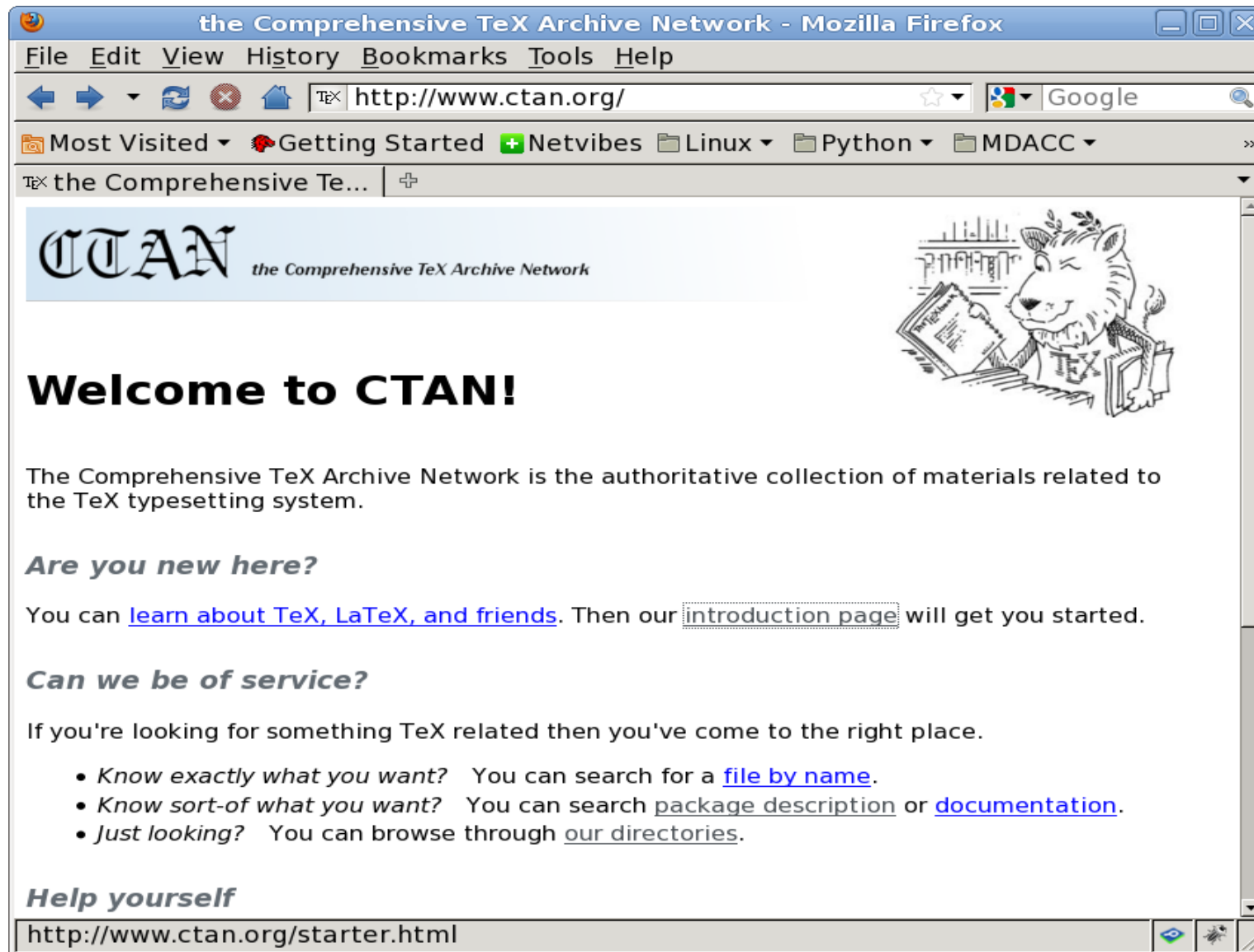
L^AT_EX is a document preparation system for high-quality typesetting.

L^AT_EX is **not** a word processor.

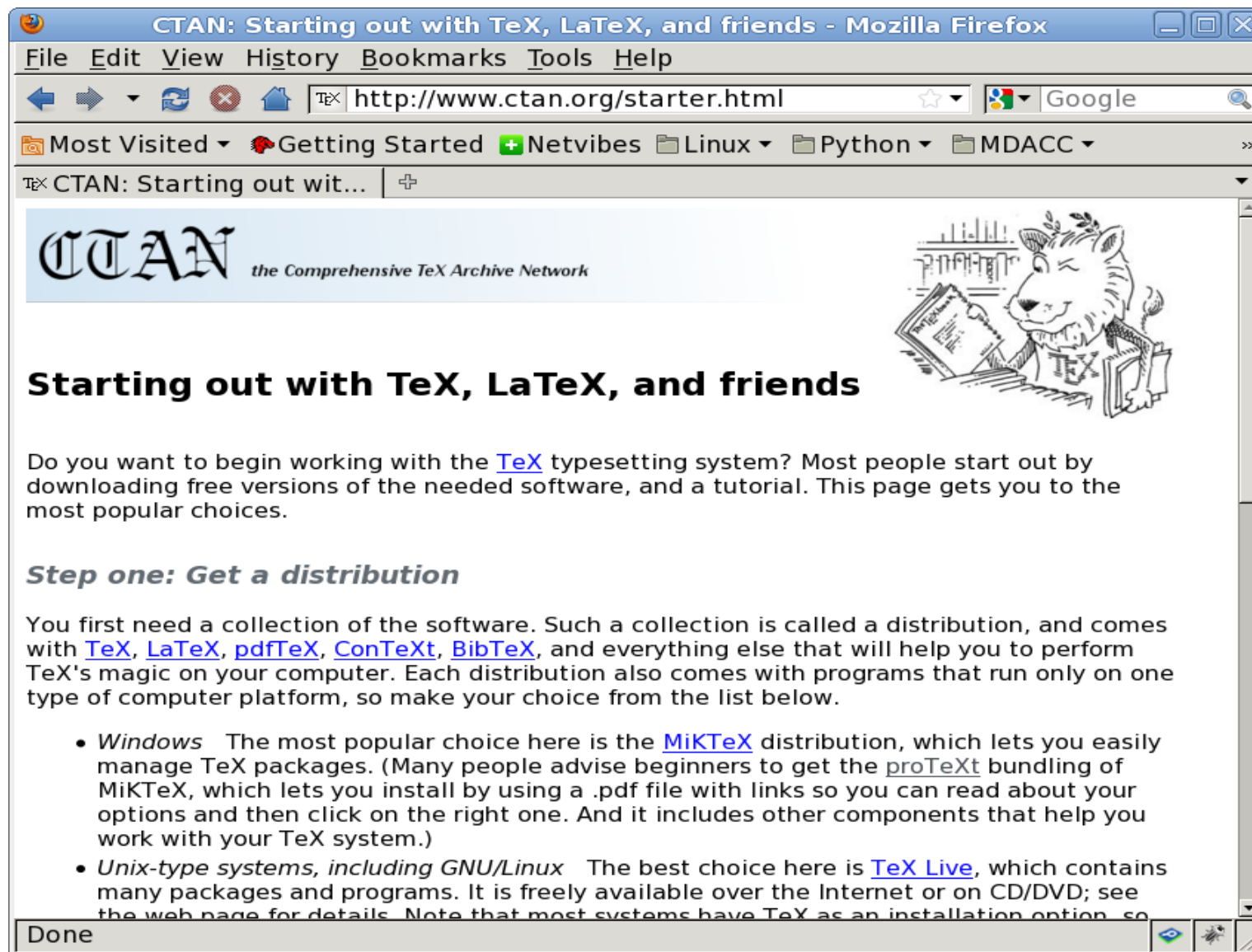
L^AT_EX separates document **content** (written by author) from **layout** (written by document designers).

You can read more about L^AT_EX at the website for the Comprehensive TeX Archive Network (CTAN): <http://www.ctan.org>.

CTAN Website



CTAN Starting Out



CTAN: Starting out with TeX, LaTeX, and friends - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://www.ctan.org/starter.html

Most Visited Getting Started Netvibes Linux Python MDACC

CTAN: Starting out wit...

CTAN the Comprehensive TeX Archive Network

Starting out with TeX, LaTeX, and friends

Do you want to begin working with the [TeX](#) typesetting system? Most people start out by downloading free versions of the needed software, and a tutorial. This page gets you to the most popular choices.

Step one: Get a distribution

You first need a collection of the software. Such a collection is called a distribution, and comes with [TeX](#), [LaTeX](#), [pdfTeX](#), [ConTeXt](#), [BibTeX](#), and everything else that will help you to perform TeX's magic on your computer. Each distribution also comes with programs that run only on one type of computer platform, so make your choice from the list below.

- **Windows** The most popular choice here is the [MiKTeX](#) distribution, which lets you easily manage TeX packages. (Many people advise beginners to get the [proTeXt](#) bundling of MiKTeX, which lets you install by using a .pdf file with links so you can read about your options and then click on the right one. And it includes other components that help you work with your TeX system.)
- **Unix-type systems, including GNU/Linux** The best choice here is [TeX Live](#), which contains many packages and programs. It is freely available over the Internet or on CD/DVD; see the web page for details. Note that most systems have TeX as an installation option, so

Done

Installing T_EX

The standard version of T_EX or L^AT_EX for Windows is MiKTeX, which is available at <http://www.miktex.org>. The current version is 2.8.

Follow the **MiKTeX** link, and then choose Download MiKTeX 2.8 from the panel on the left.

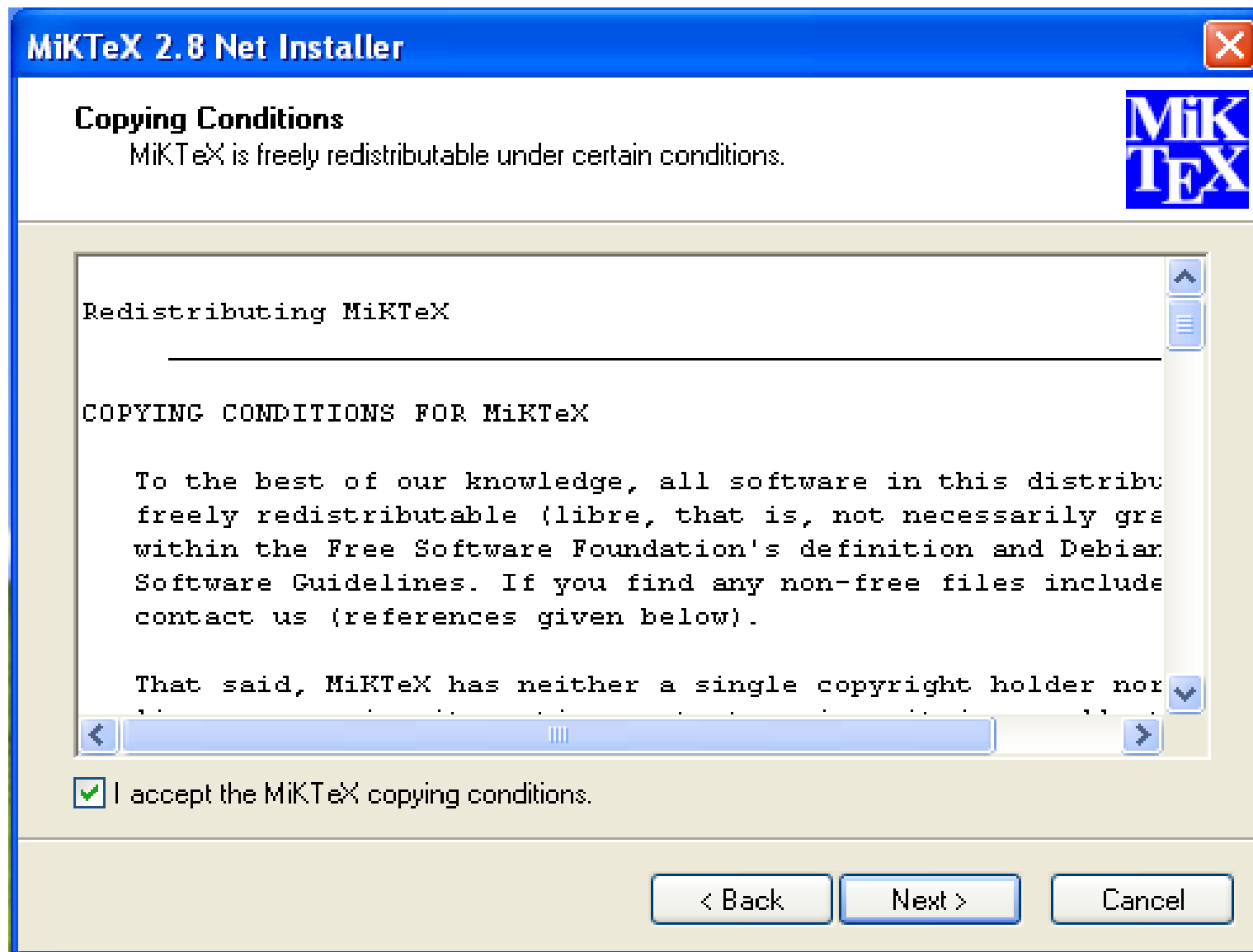
I used the “MikTeX 2.8 Net Installer” because it’s relatively small (3 MB), but ran into some problems adding additional packages.

It might be better to download the “Basic MiKTeX 2.8” installer (92 MB).

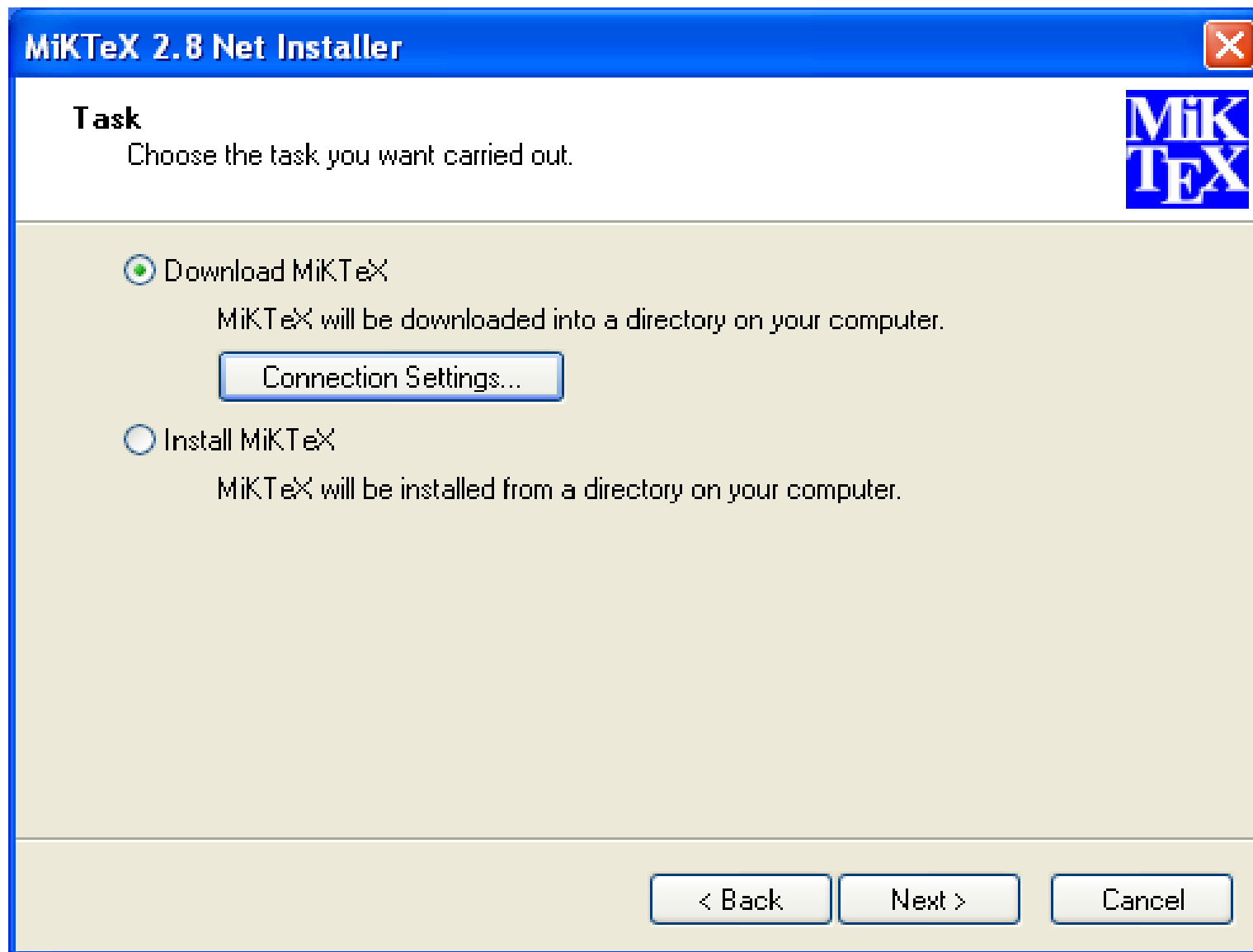
Keep track of where you save this file (your desktop will work just fine) and then double-click on the resulting icon to start the installation.

CTAN states that the standard version of L^AT_EX for Macintosh computers is **MacTeX**. (Since I have never installed this version, you will have to figure out how to install it yourself)

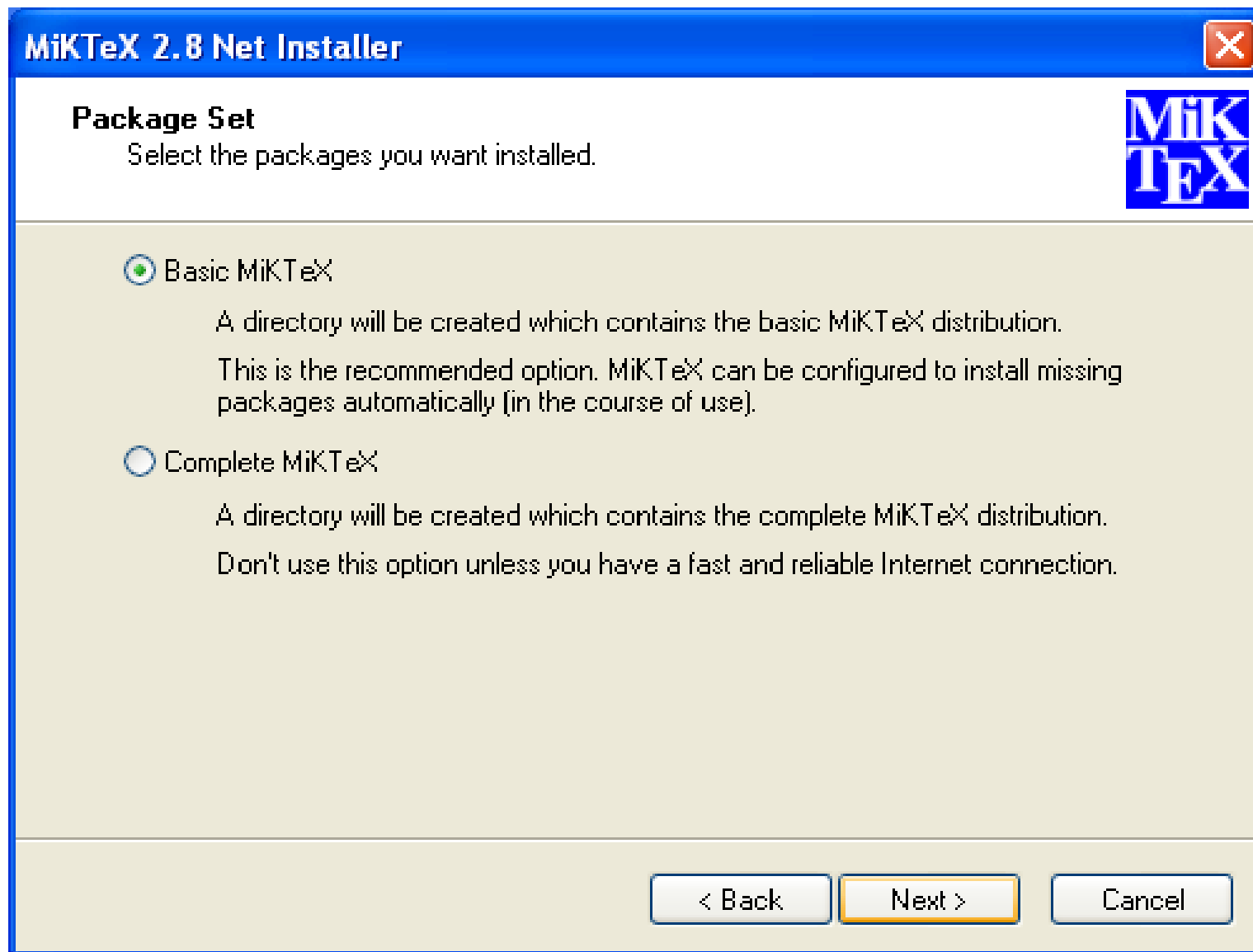
The MiKTeX Installer: License



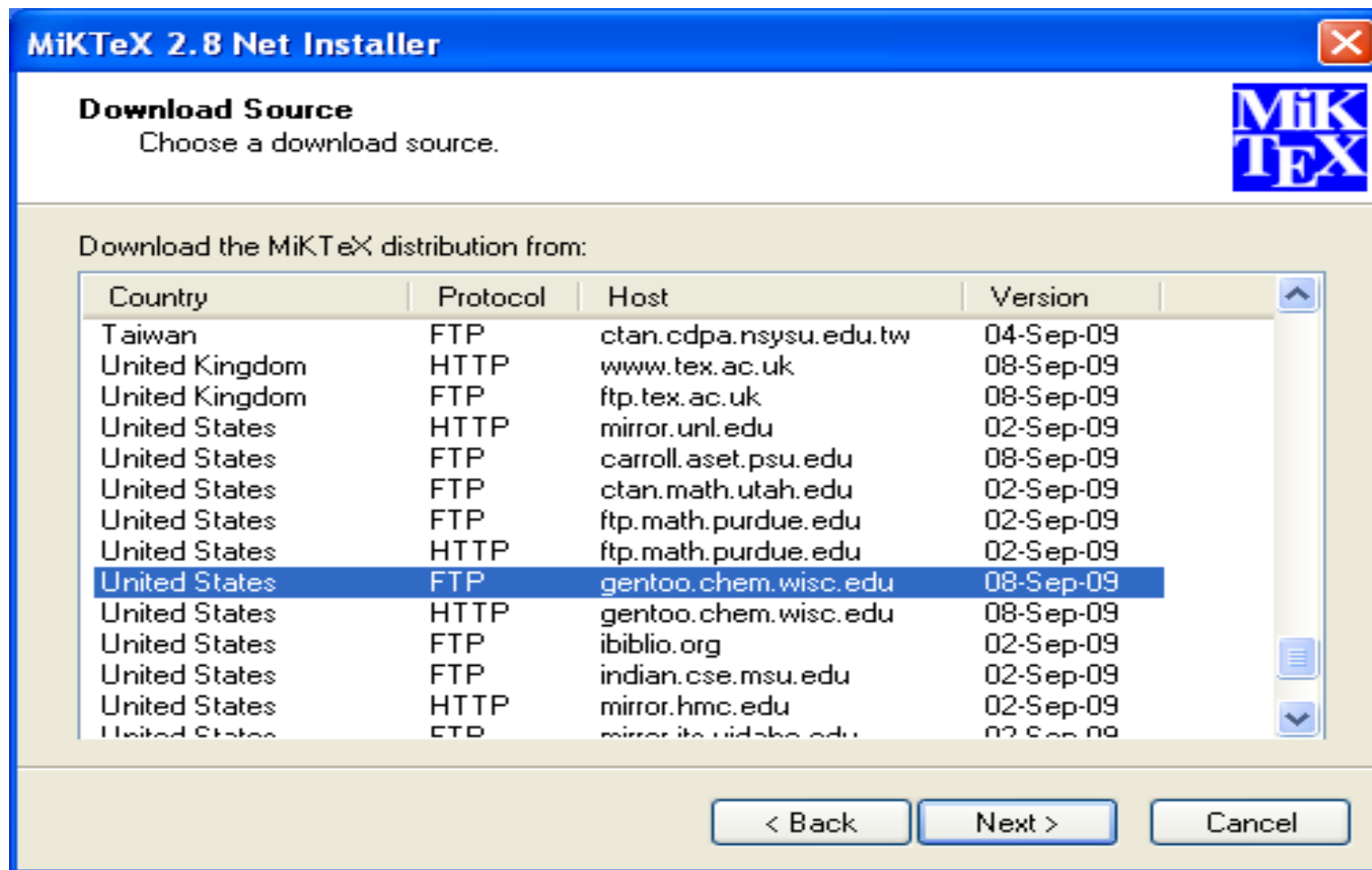
The MiKTeX Installer: Download



The MiKTeX Installer: Package Set

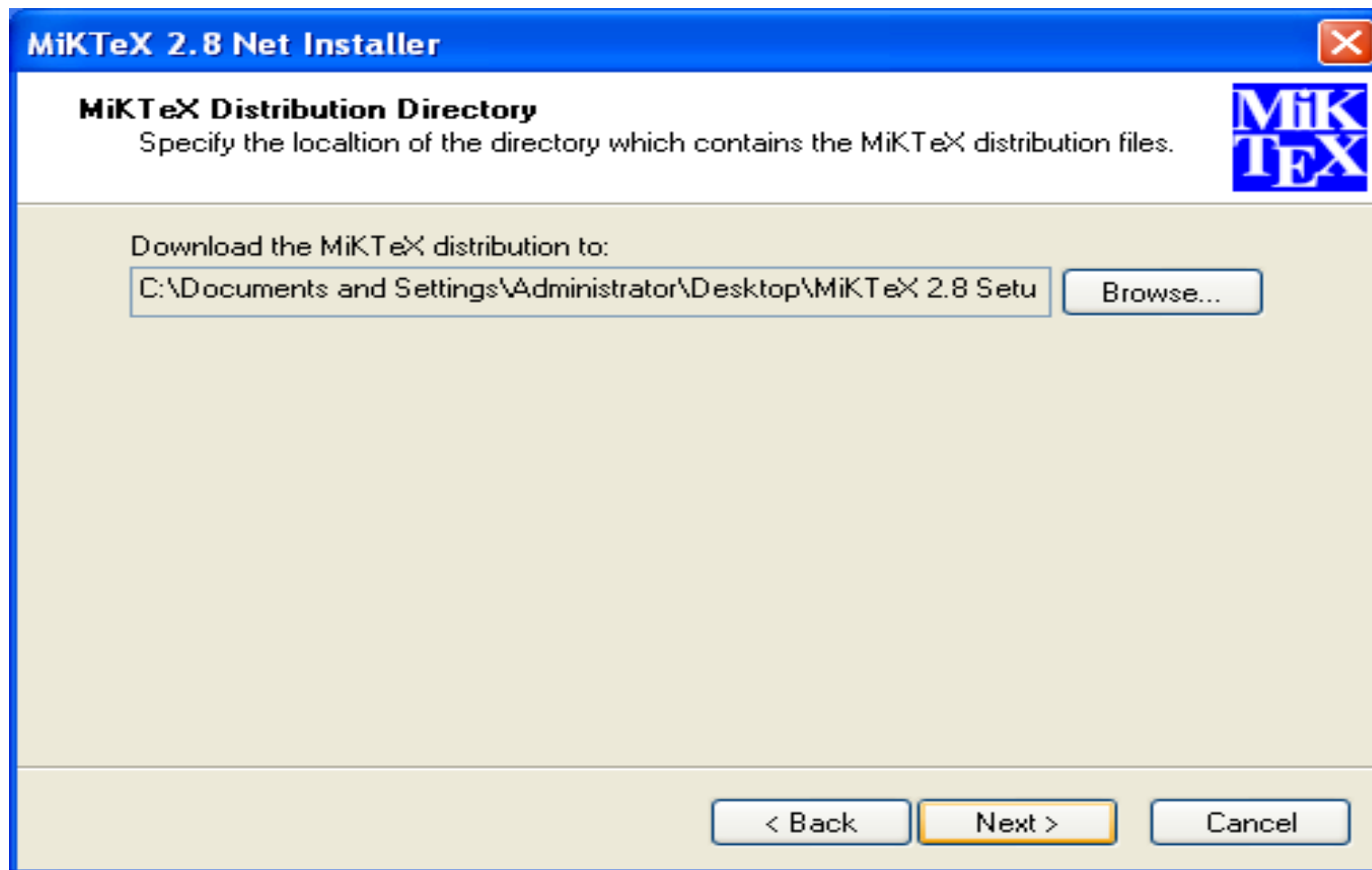


The MiKTeX Installer: Download Source



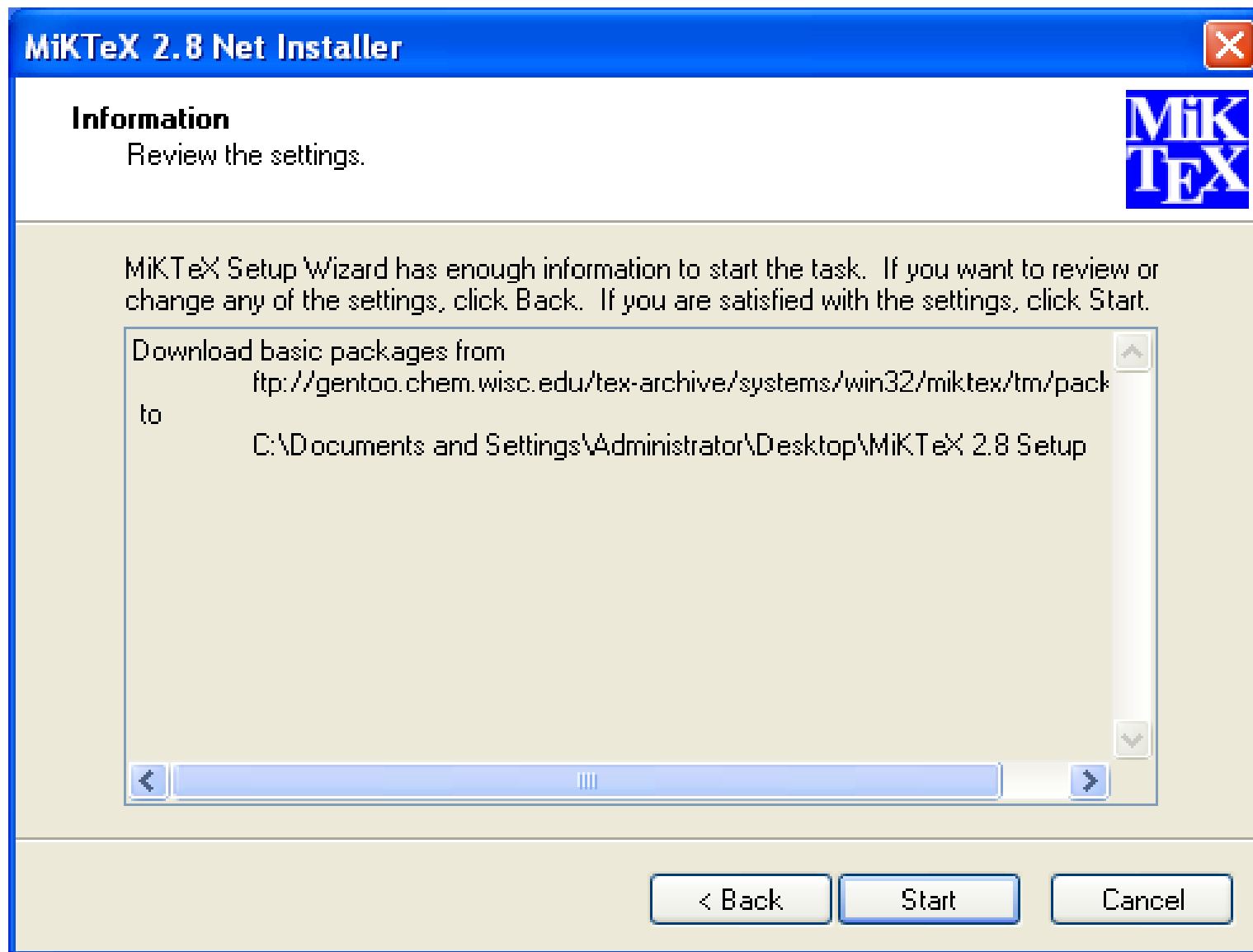
Choose an up-to-date US mirror.

The MiKTeX Installer: Distribution Directory

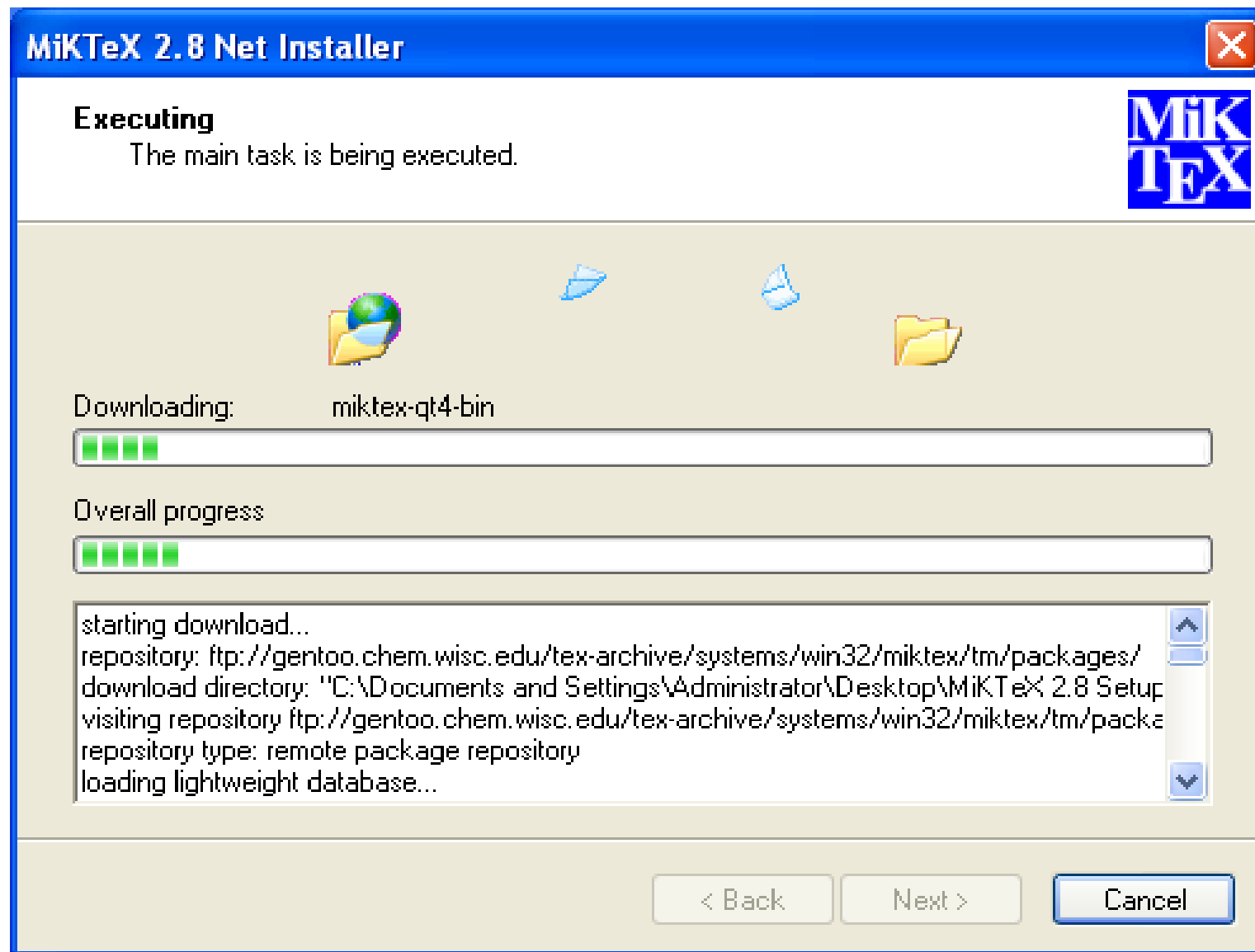


Remember this directory for later.

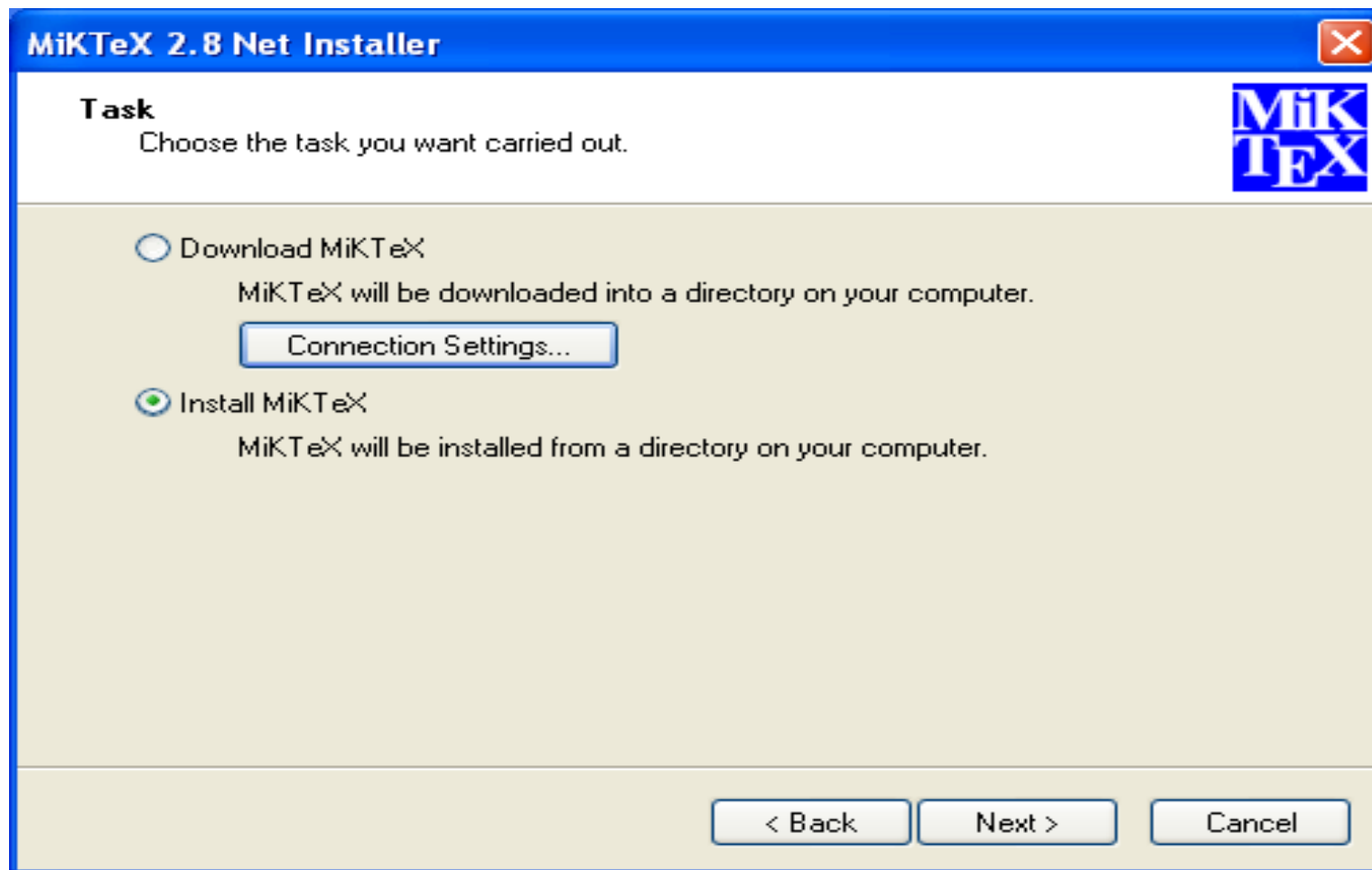
The MiKTeX Installer: Review Settings



The MiKTeX Installer: Downloading

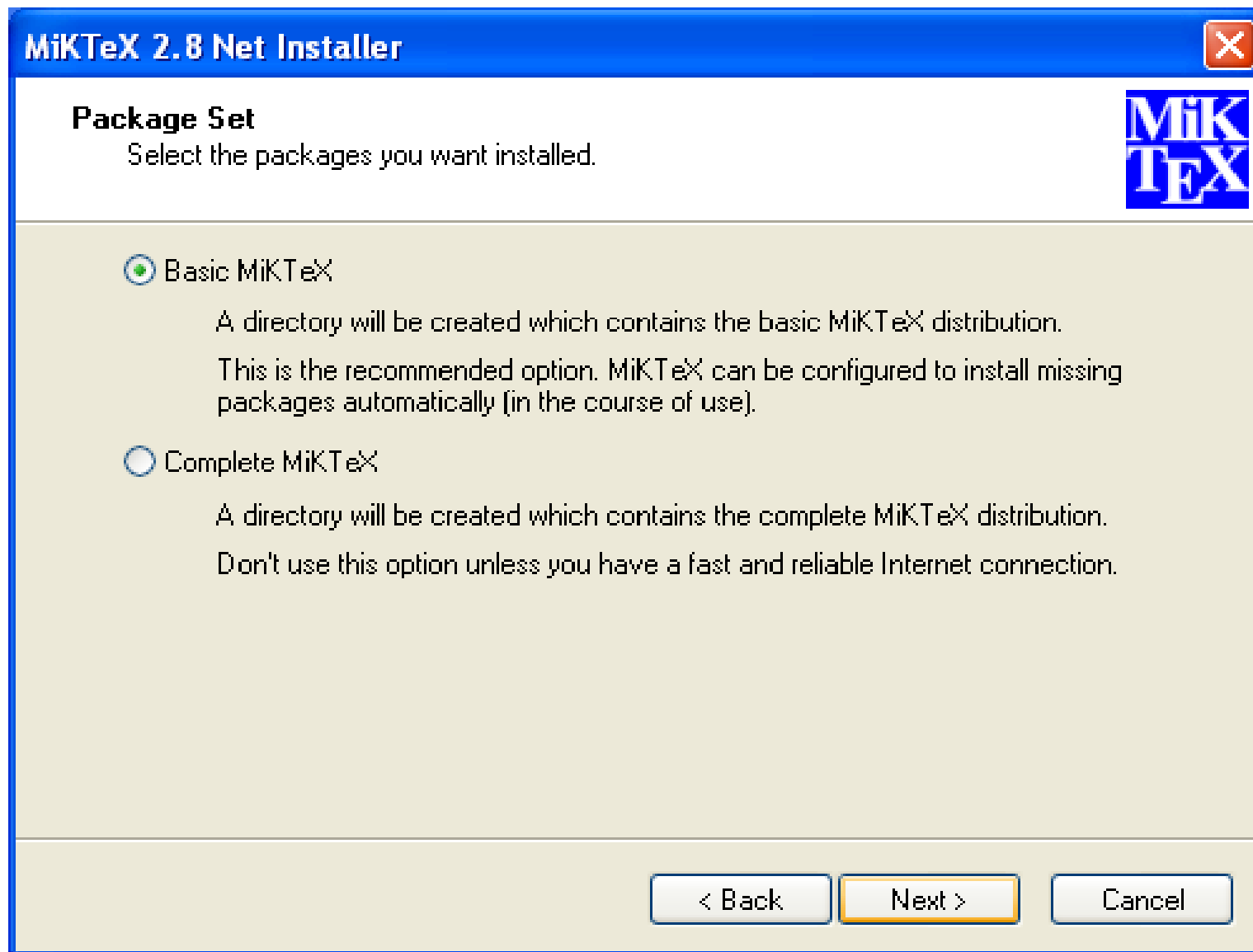


The MiKTeX Installer: Install

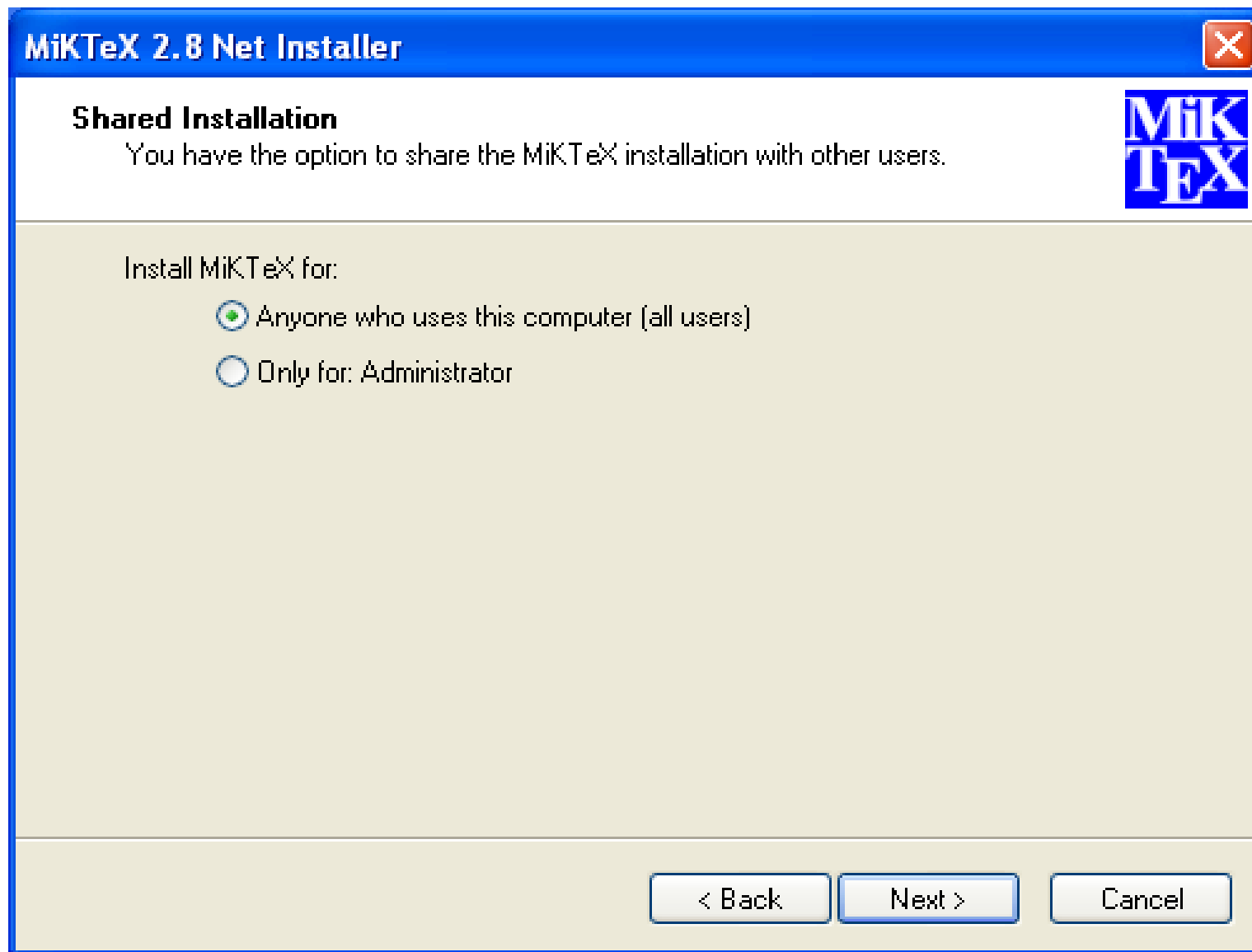


Rerun the installer, but this time choose install.

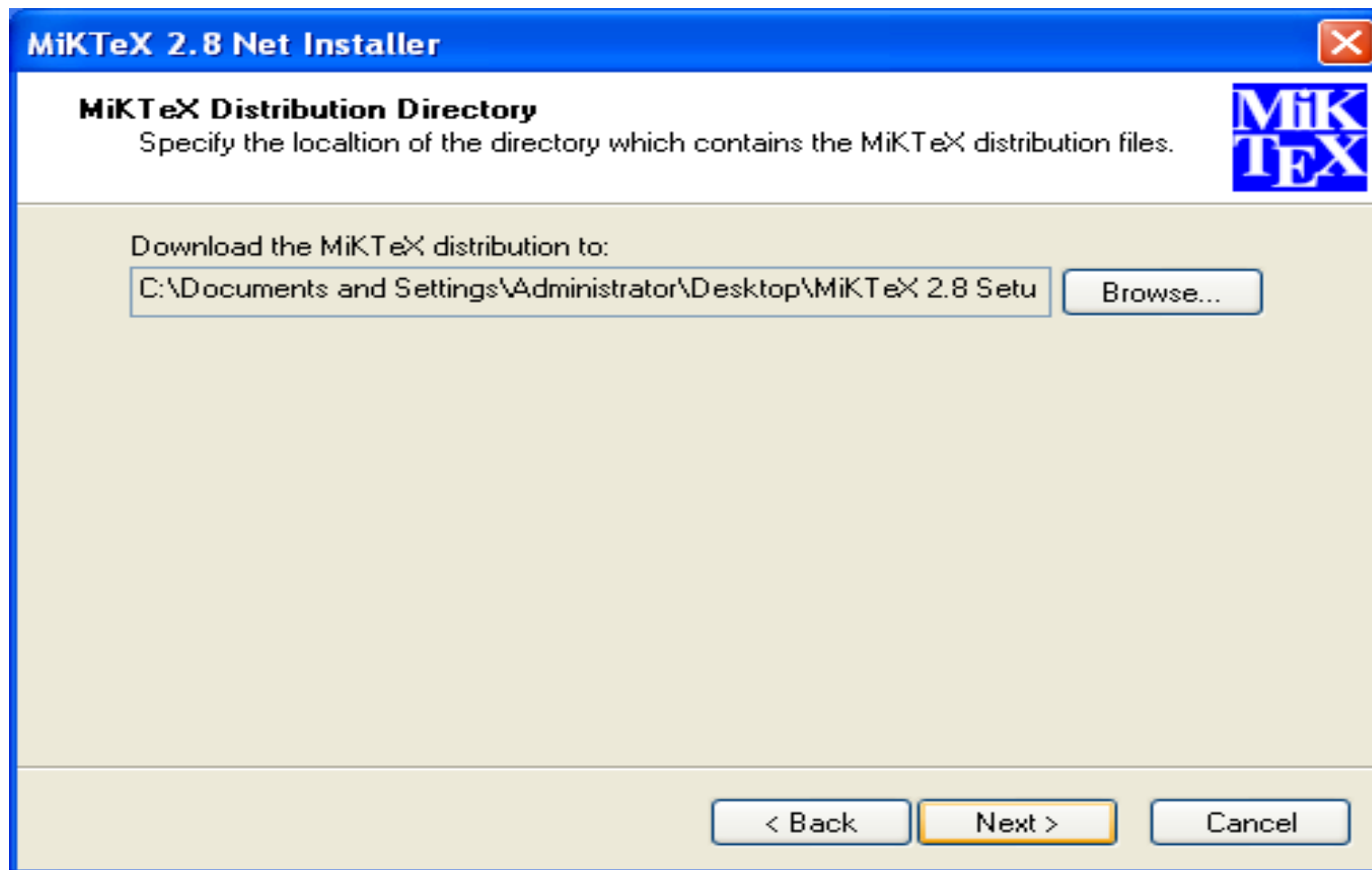
The MiKTeX Installer: Package Set



The MiKTeX Installer: Shared Install

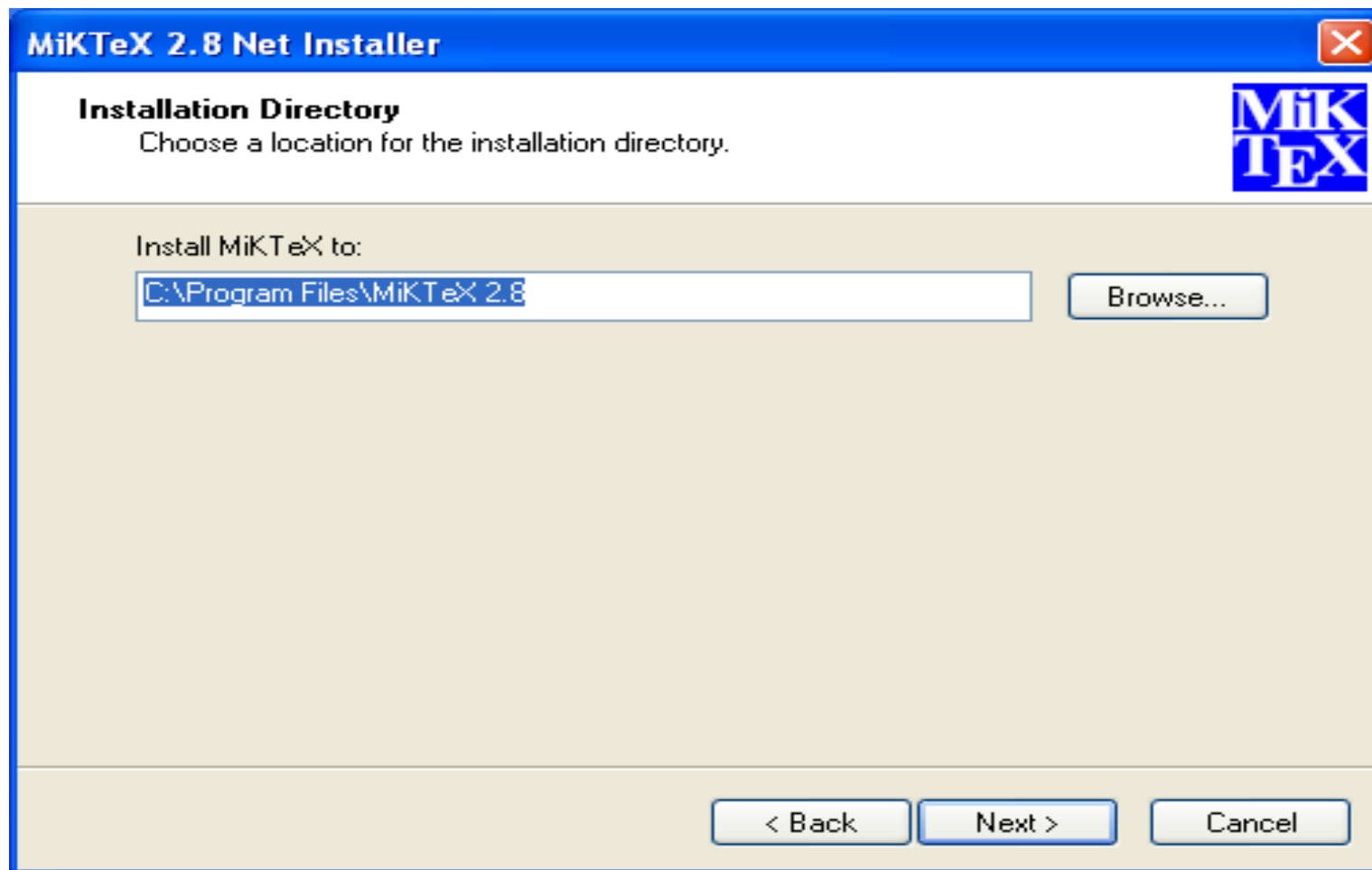


The MiKTeX Installer: Distribution Directory



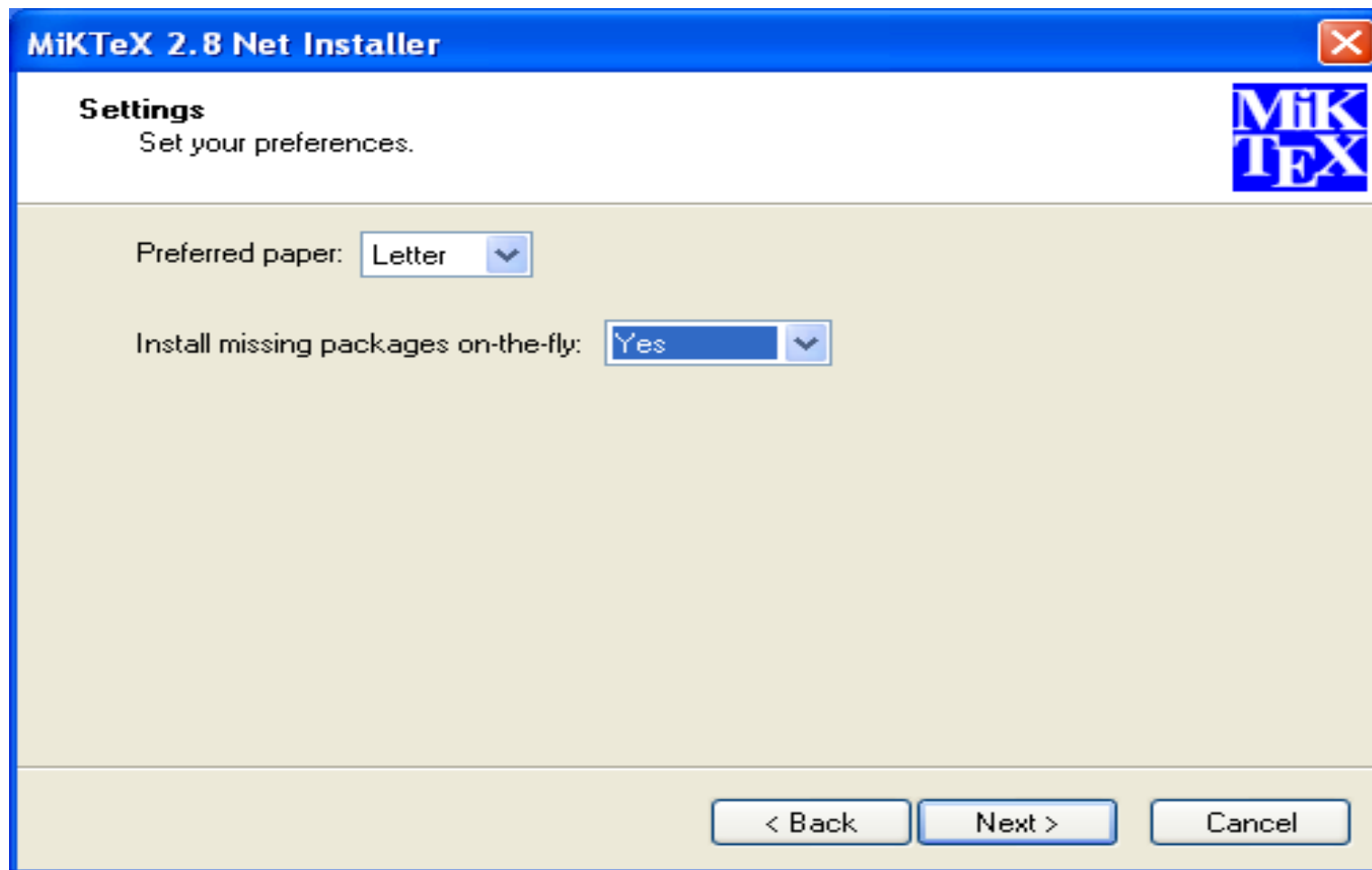
Use the directory you downloaded the distribution to earlier.

The MiKTeX Installer: Installation Directory



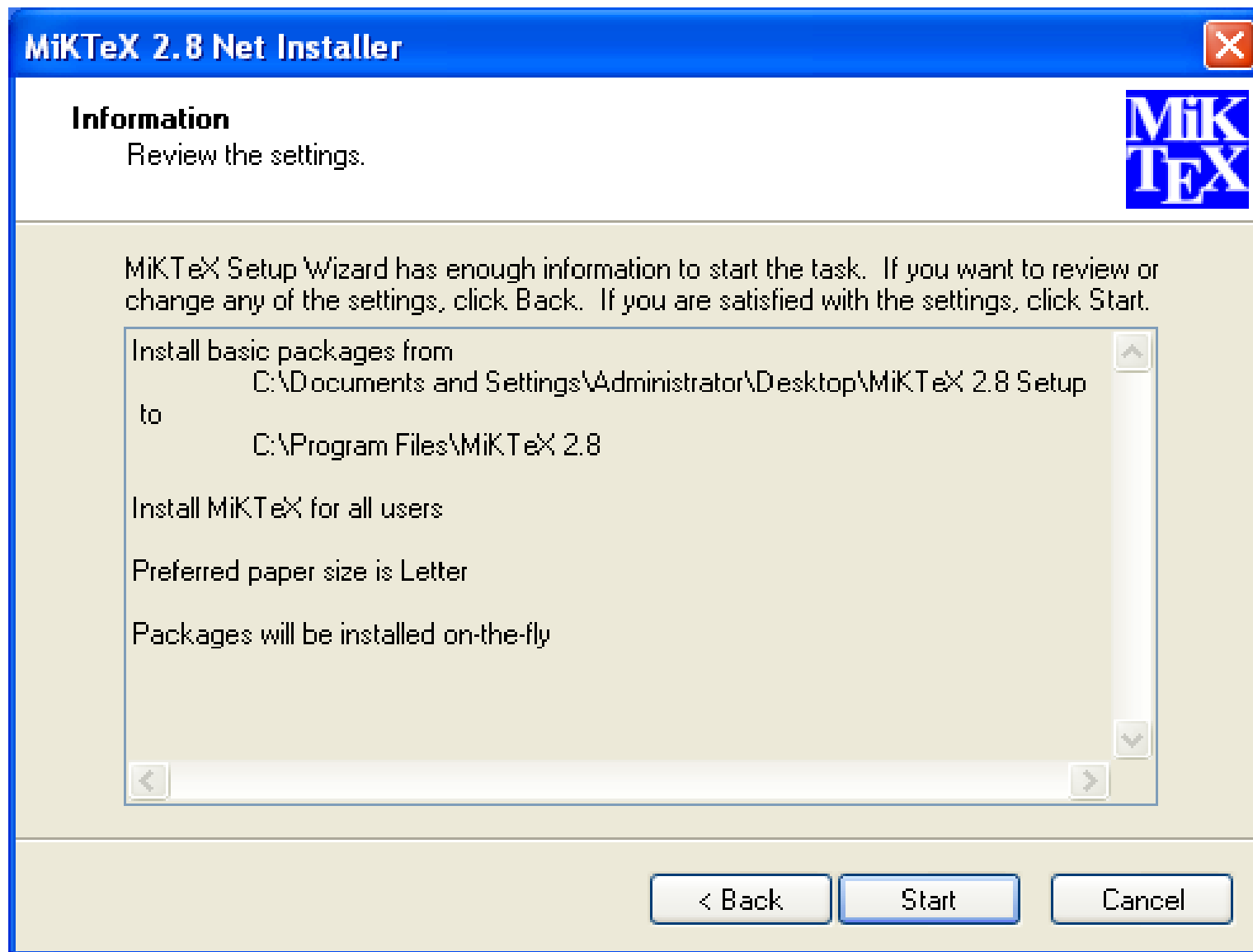
Best if you can avoid spaces in this path. Remember this path.

The MiKTeX Installer: Default Settings

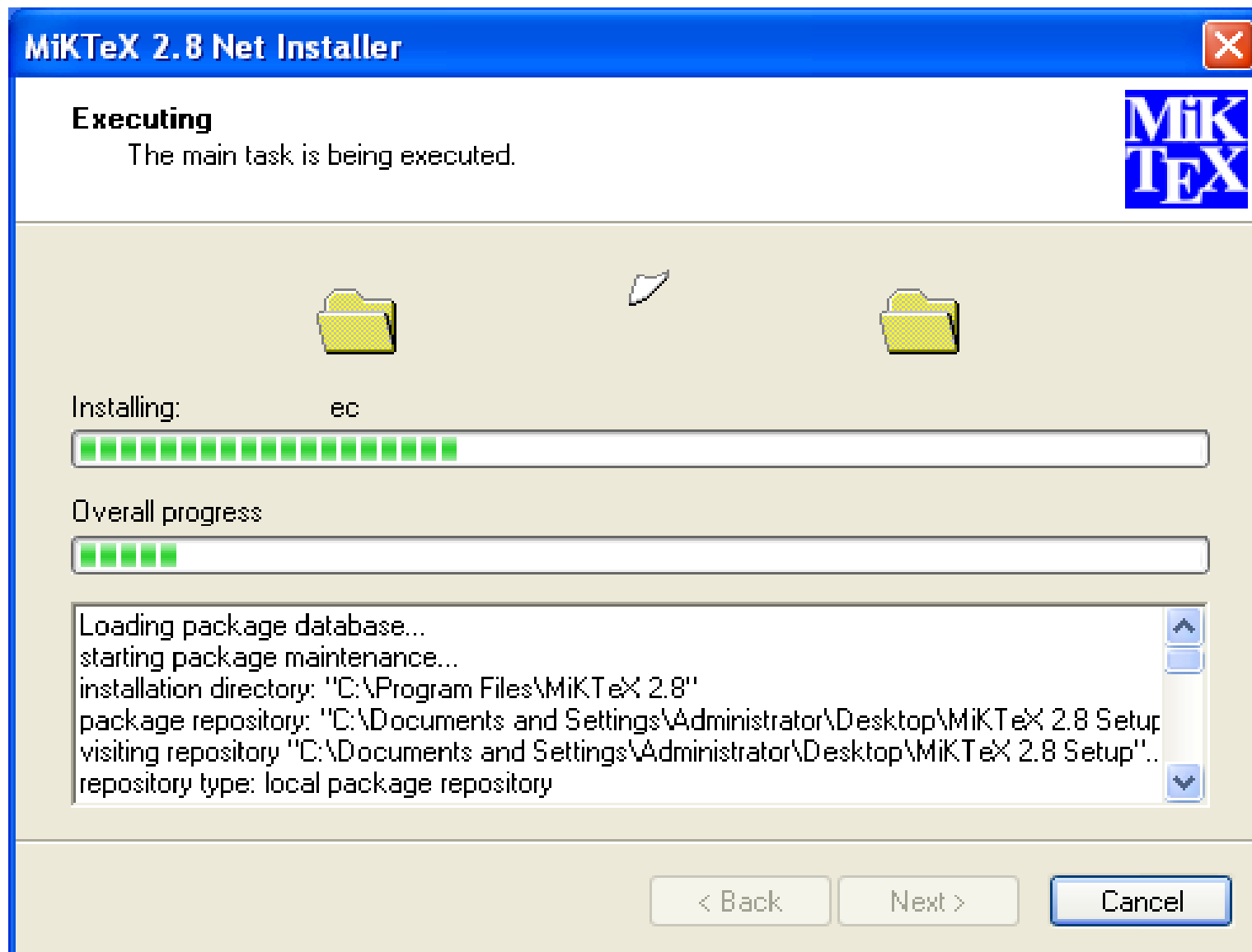


Make sure you change these settings from the defaults.

The MiKTeX Installer: Install Review



The MiKTeX Installer: Installing at last



Ghostscript and GSview

Goto <http://www.ghostscript.com> and download `gs870w32.exe` from either `cs.wisc.edu` or `sourceforge.net`.

Follow the links at the bottom of the ghostscript page to download **GSview 4.9** (`gsv49w32.exe`).

These are both straight-forward installs

TeXnicCenter

Optionally, you can download and install **TeXnicCenter**.

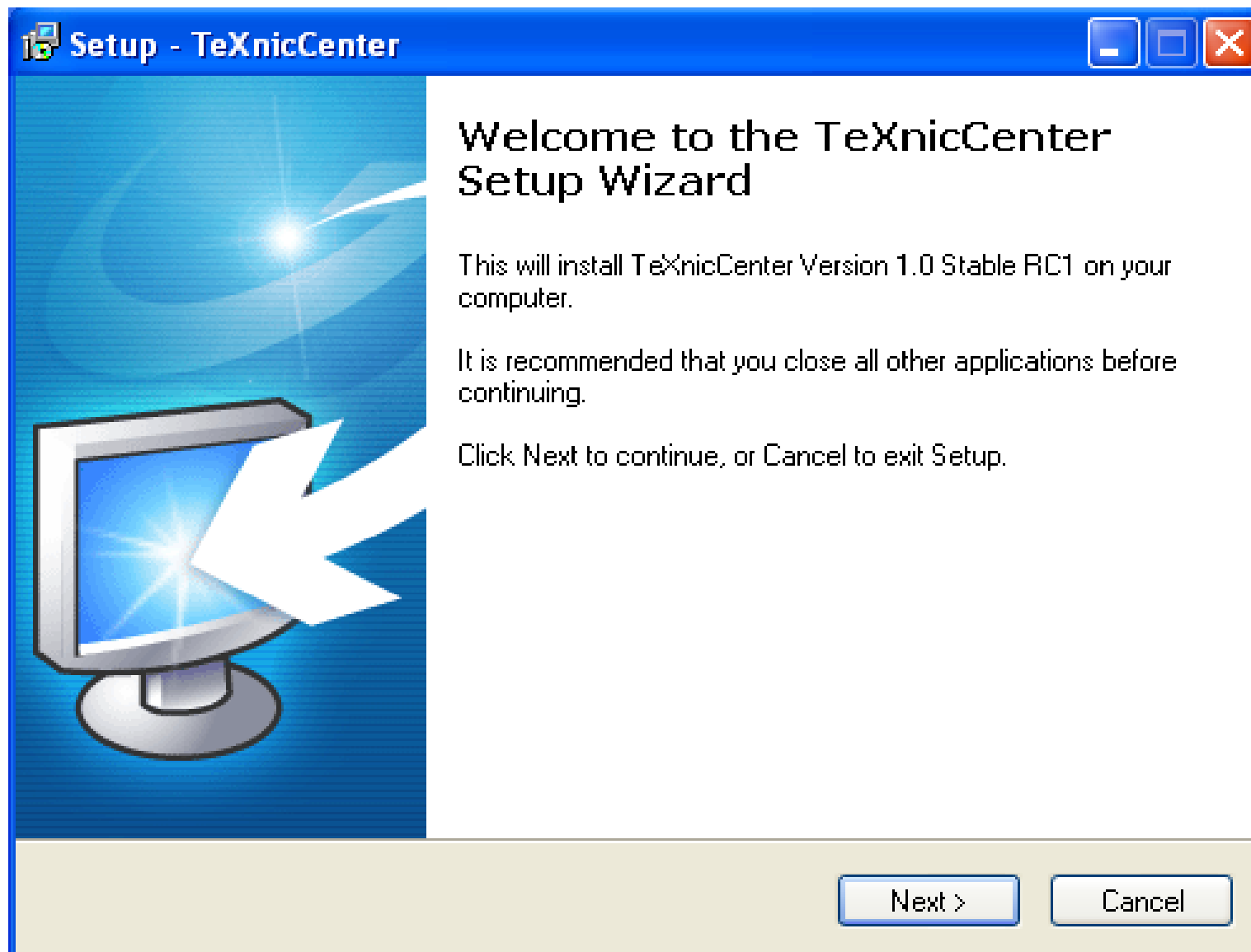
TeXnicCenter is an integrated environment for creating \LaTeX documents using Microsoft Windows.

- \LaTeX specific editor with syntax highlighting, bracket matching, etc.
- Buttons for inserting predefined \LaTeX snippets.
- Buttons for building and viewing document.

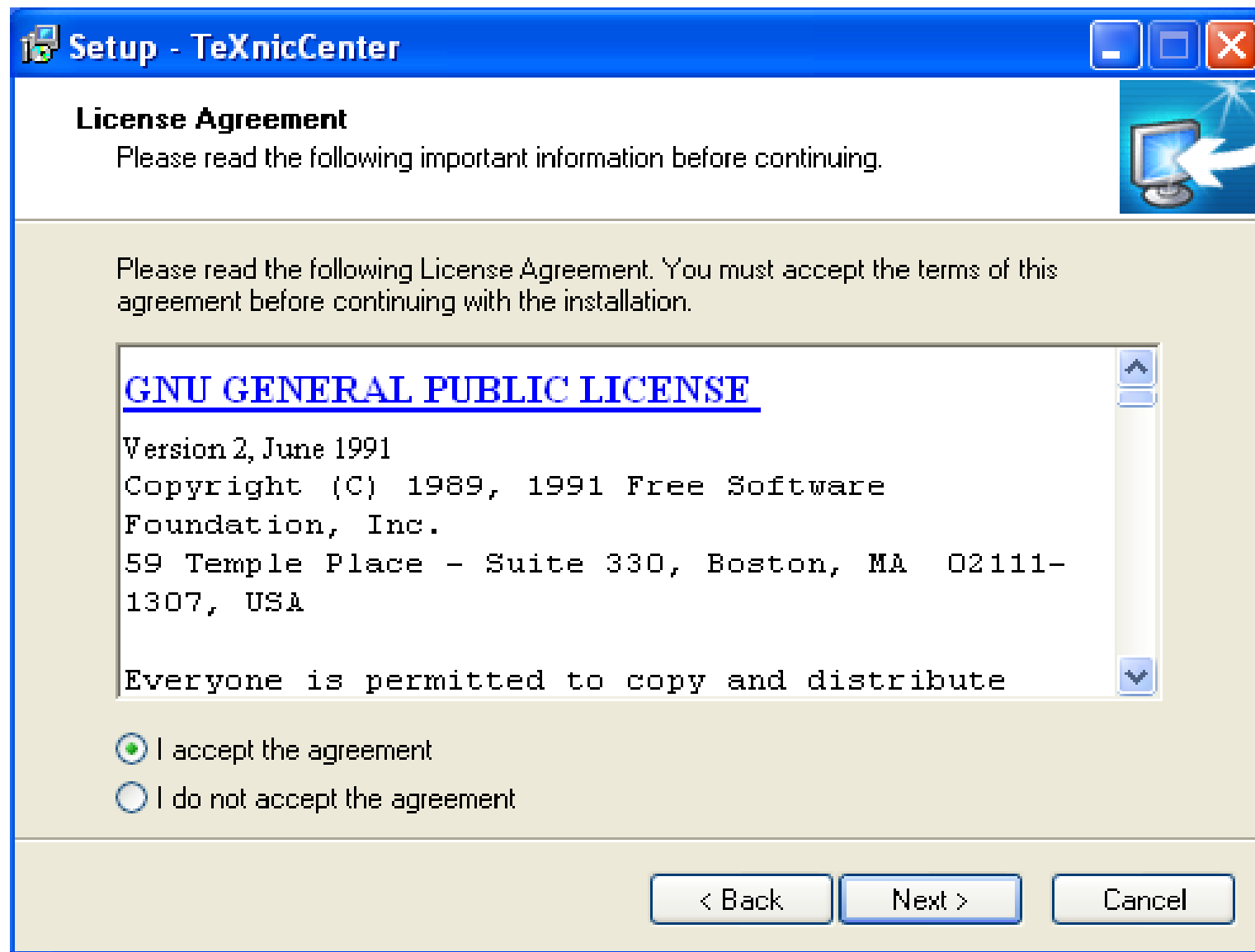
Unfortunately, **TeXnicCenter** does not know about **Sweave**, so we will still need to do some stuff manually.

Goto <http://www.texniccenter.org/> and follow the links to download the TeXnicCenter installer.

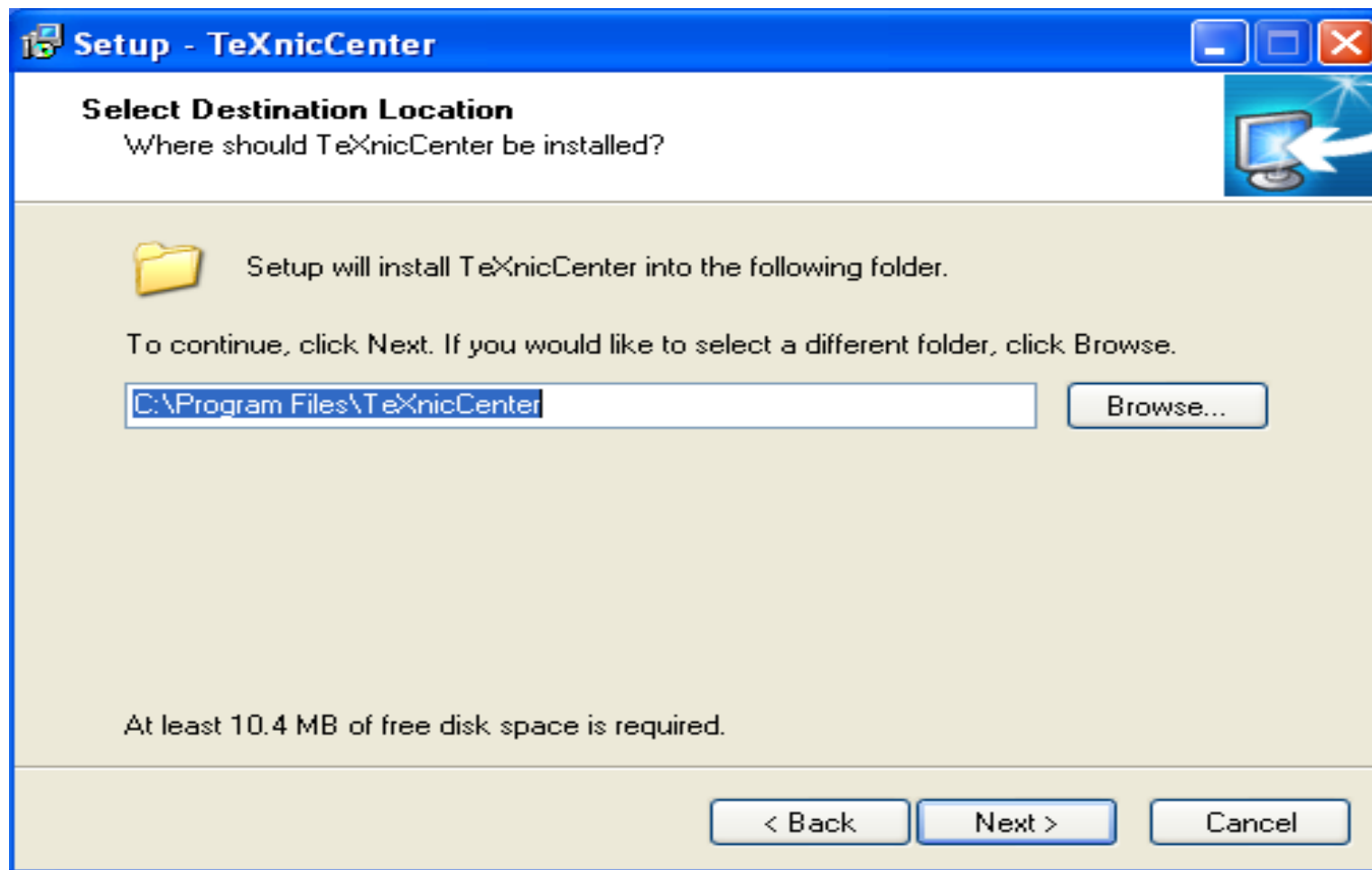
The TeXnicCenter Installer: Welcome



The TeXnicCenter Installer: License

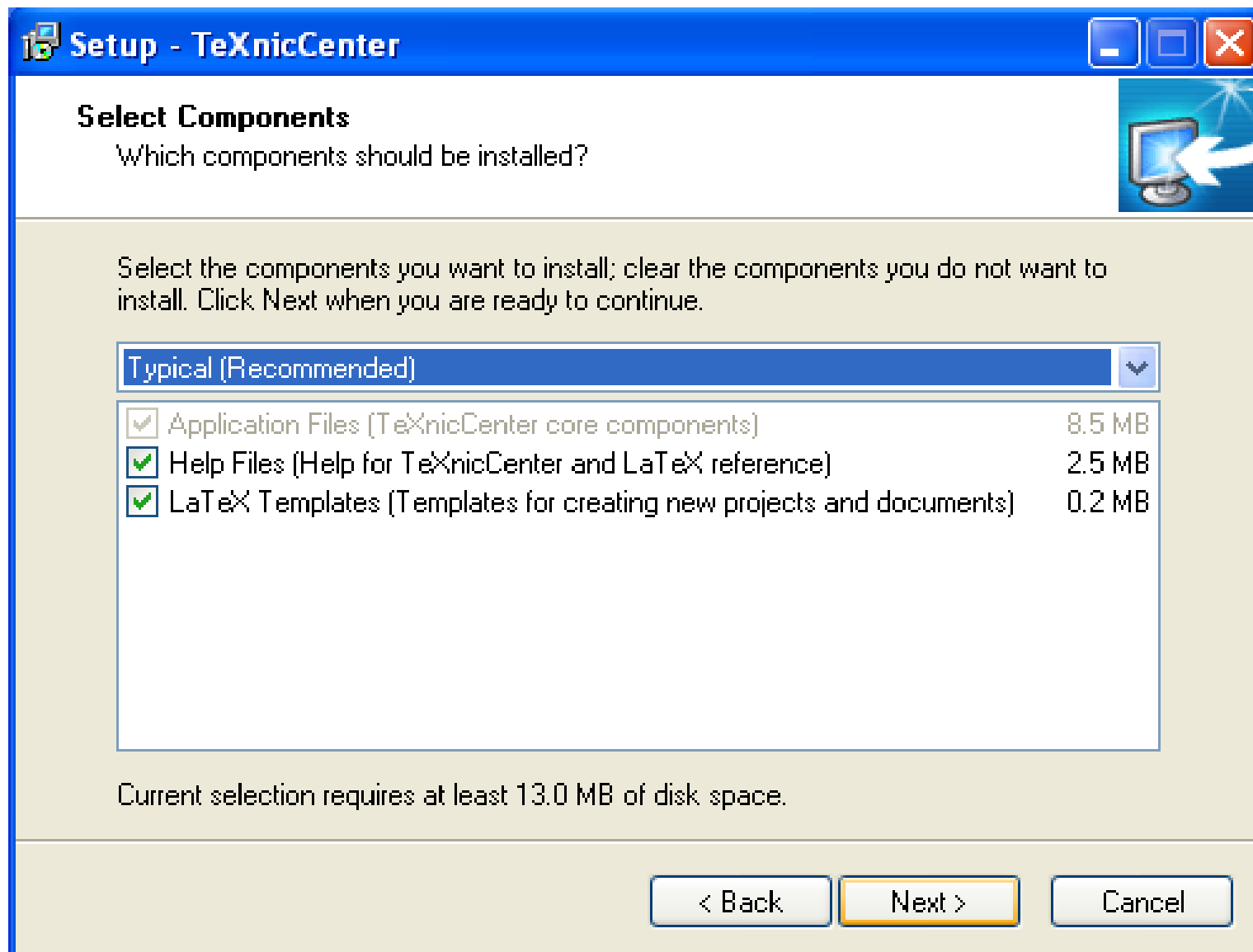


The TeXnicCenter Installer: Location

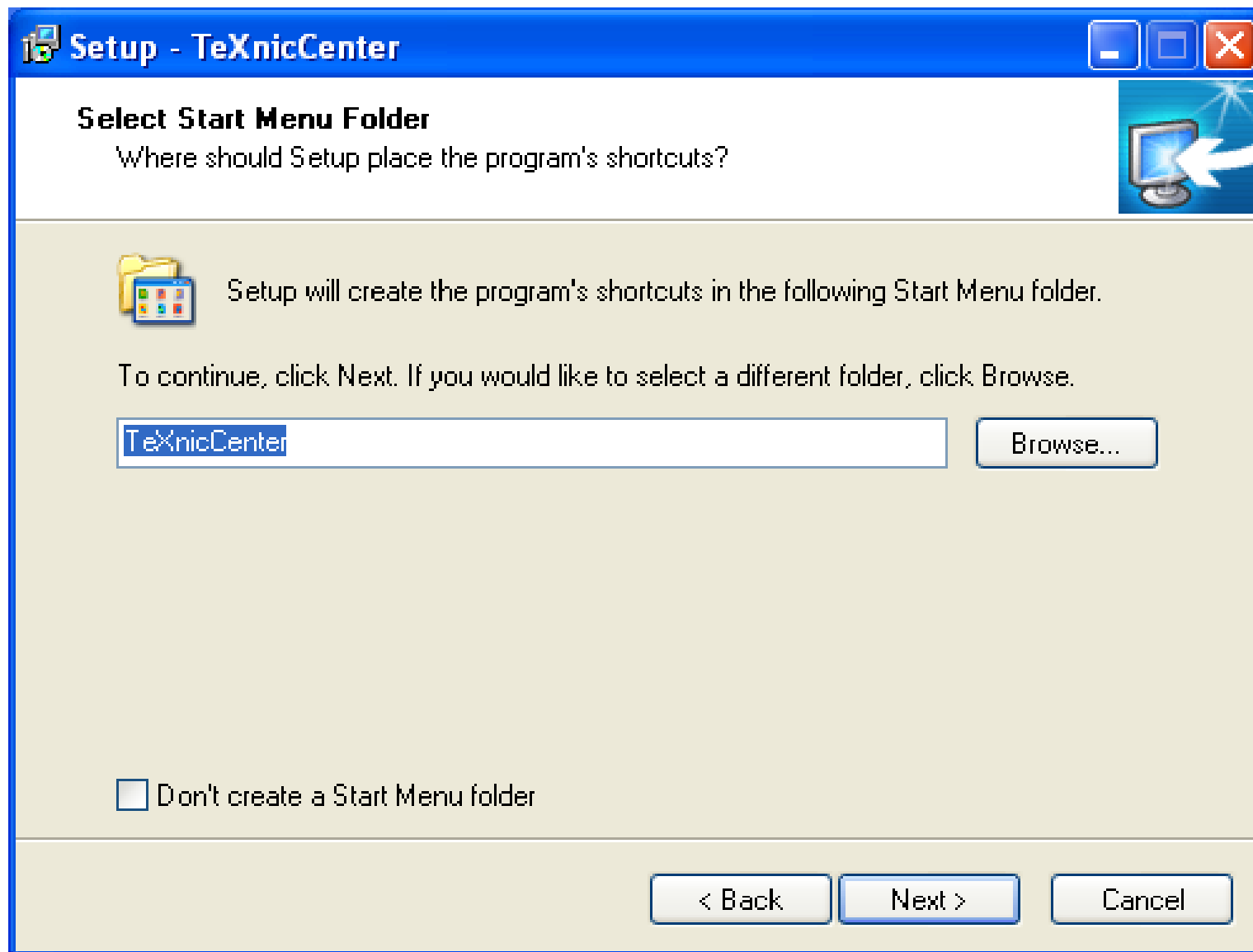


Best to avoid spaces in this path.

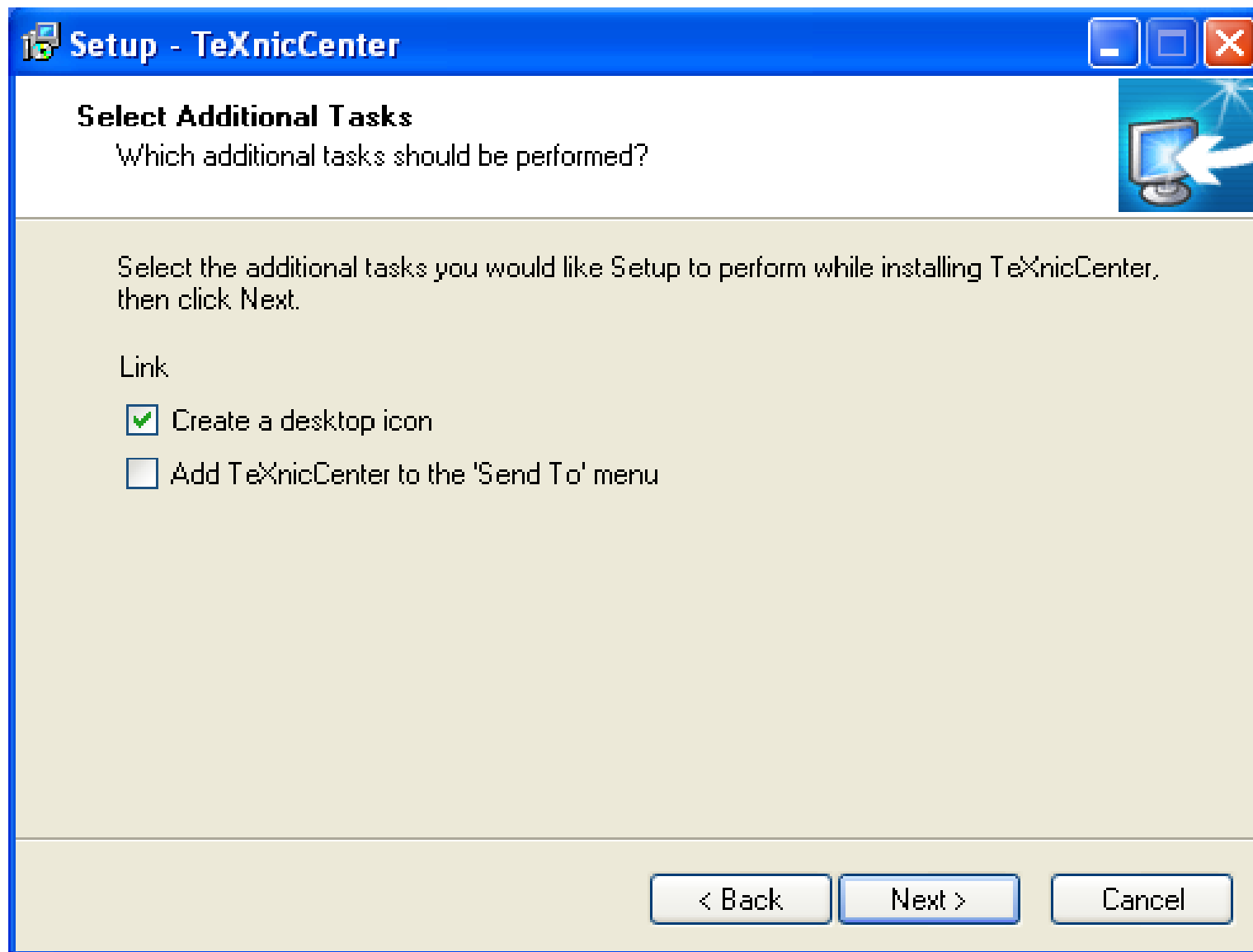
The TeXnicCenter Installer: Components



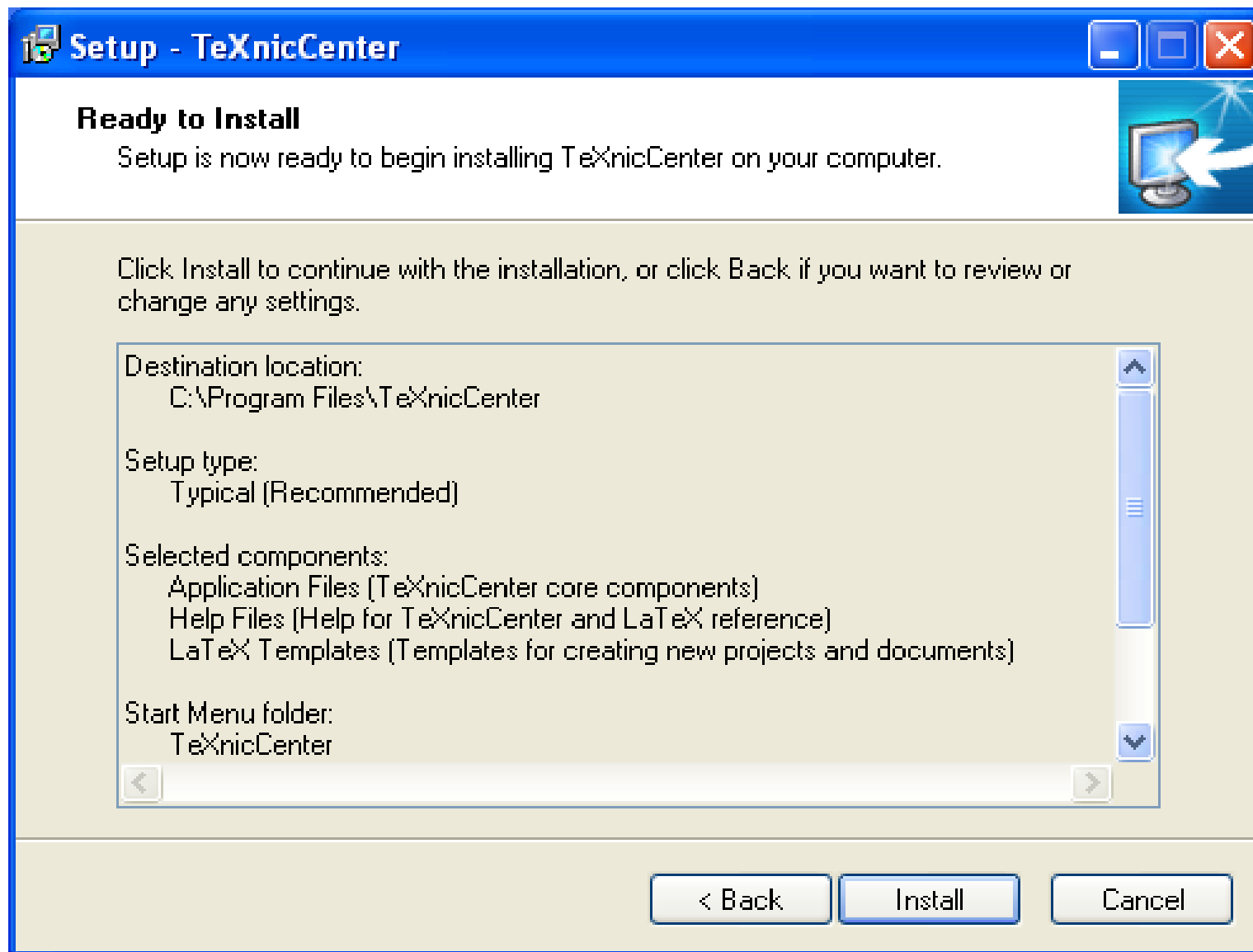
The TeXnicCenter Installer: Start Menu



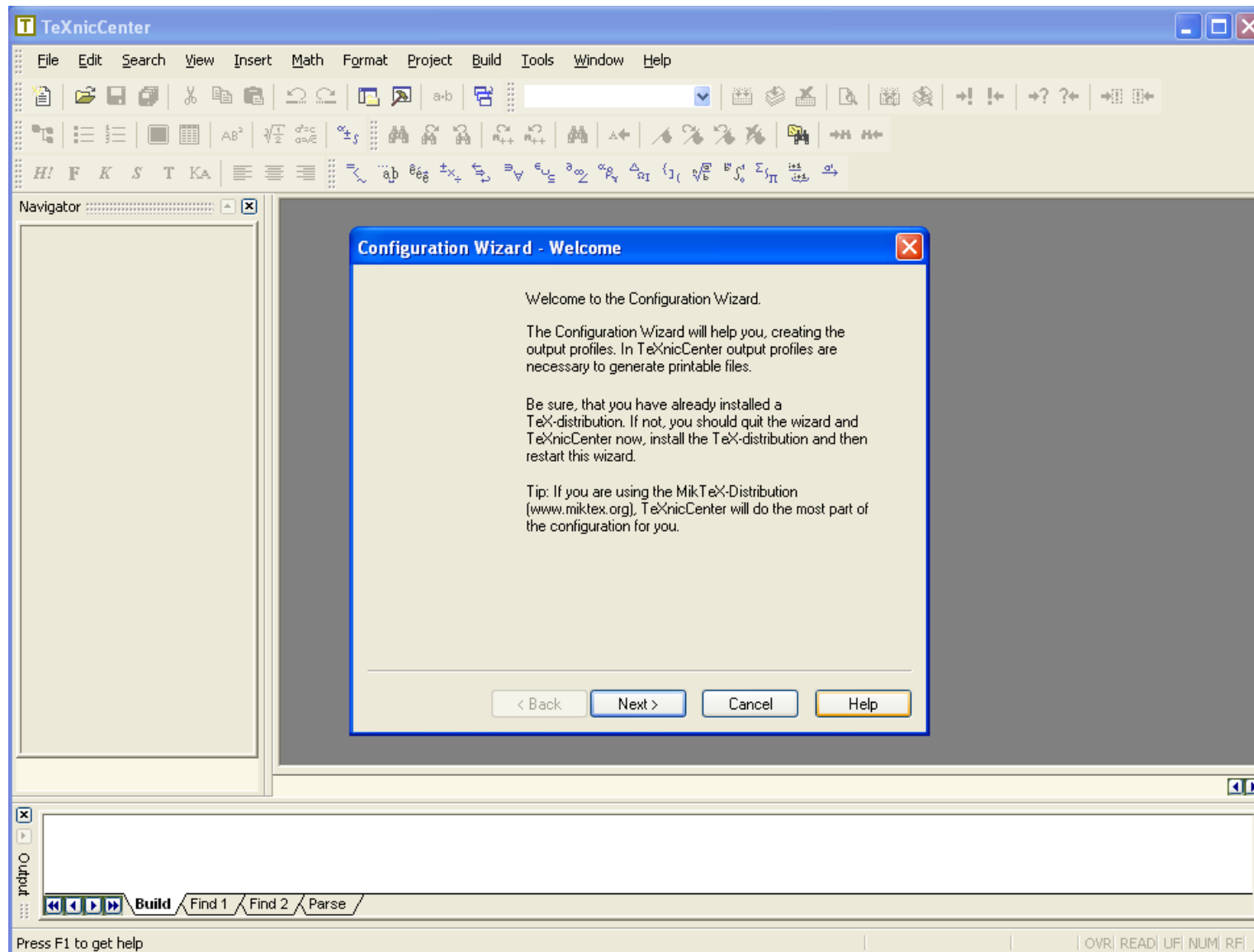
The TeXnicCenter Installer: Create Icon



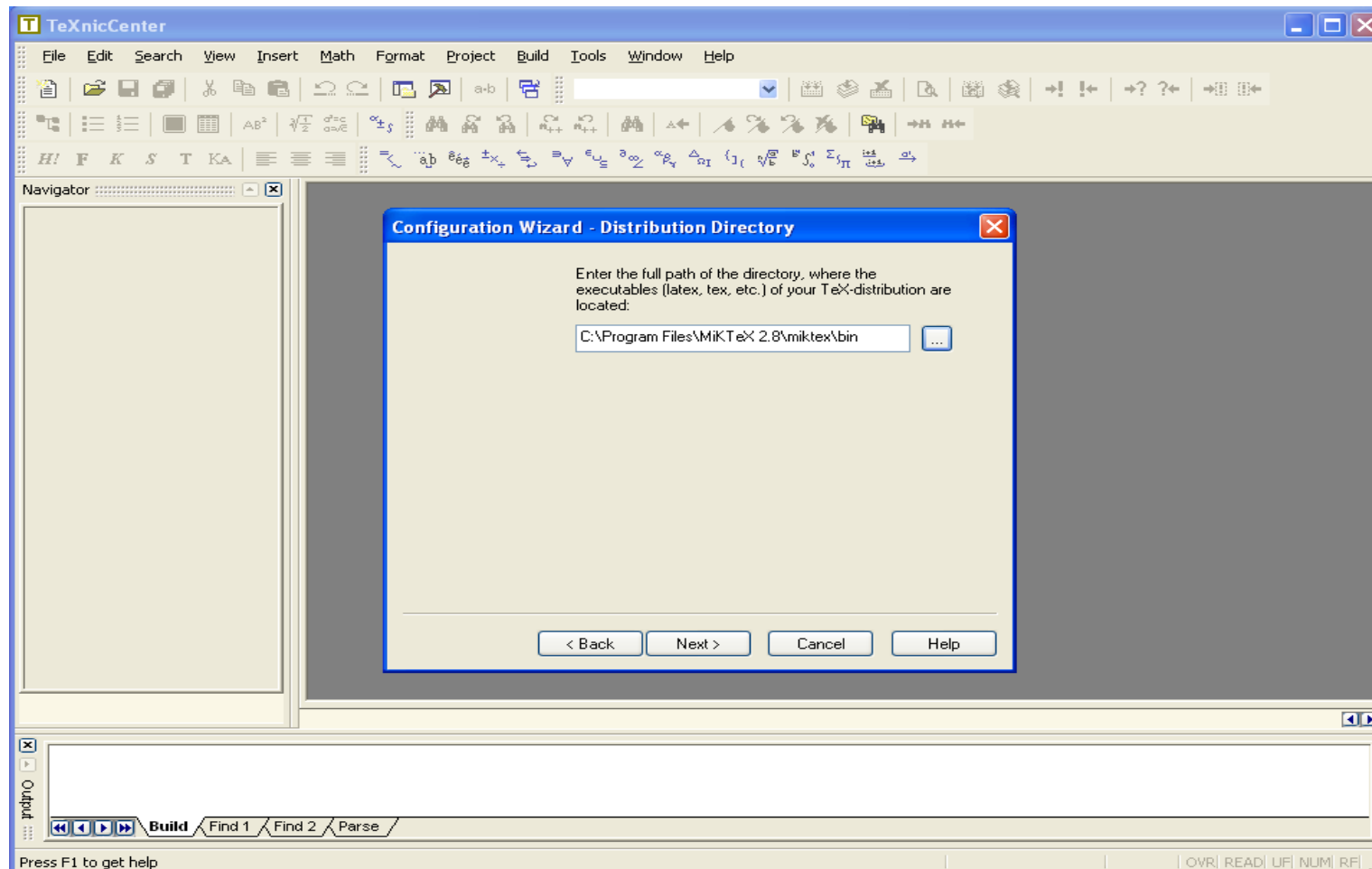
The TeXnicCenter Installer: Ready



TeXnicCenter: Config Wizard

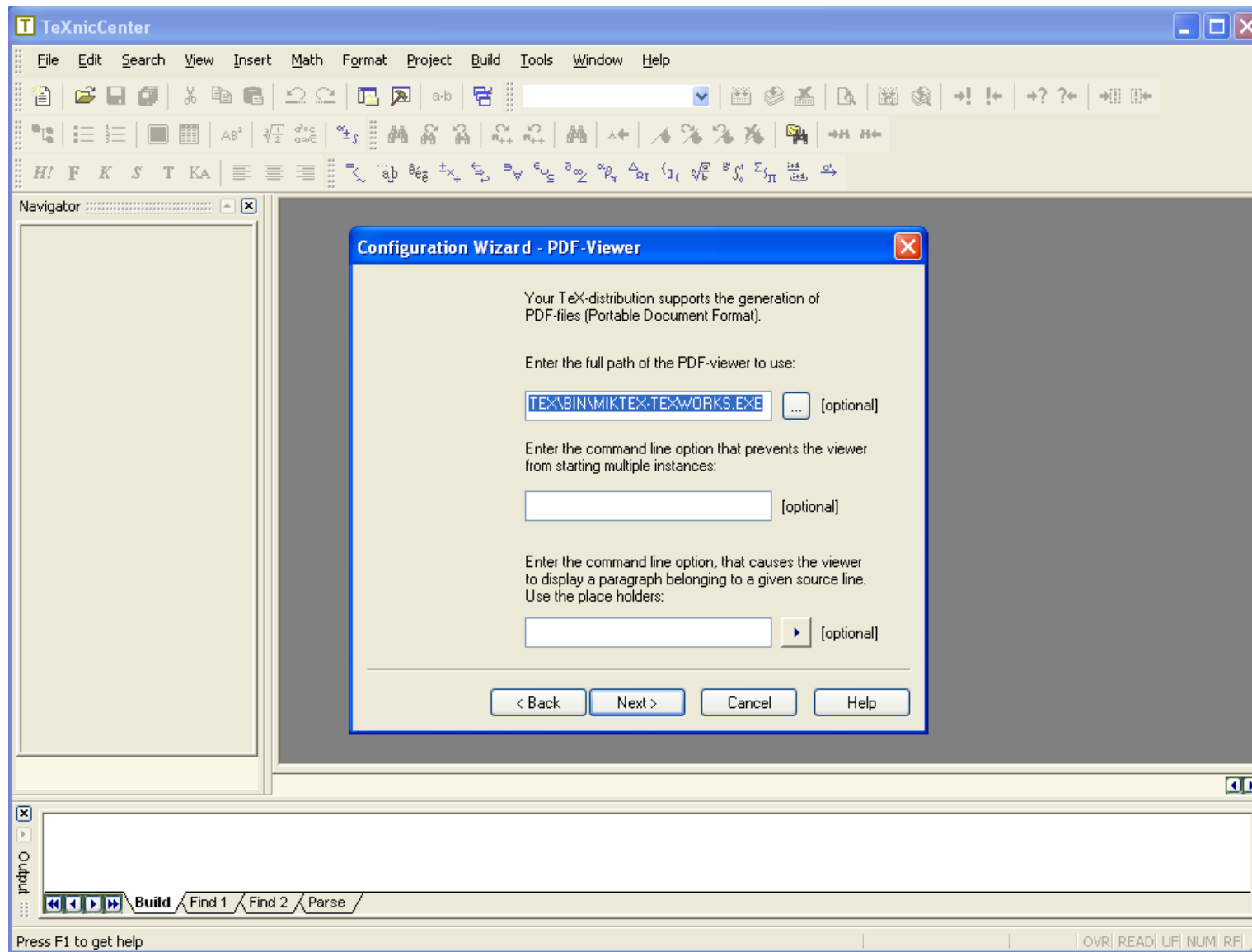


TeXnicCenter: Config Wizard

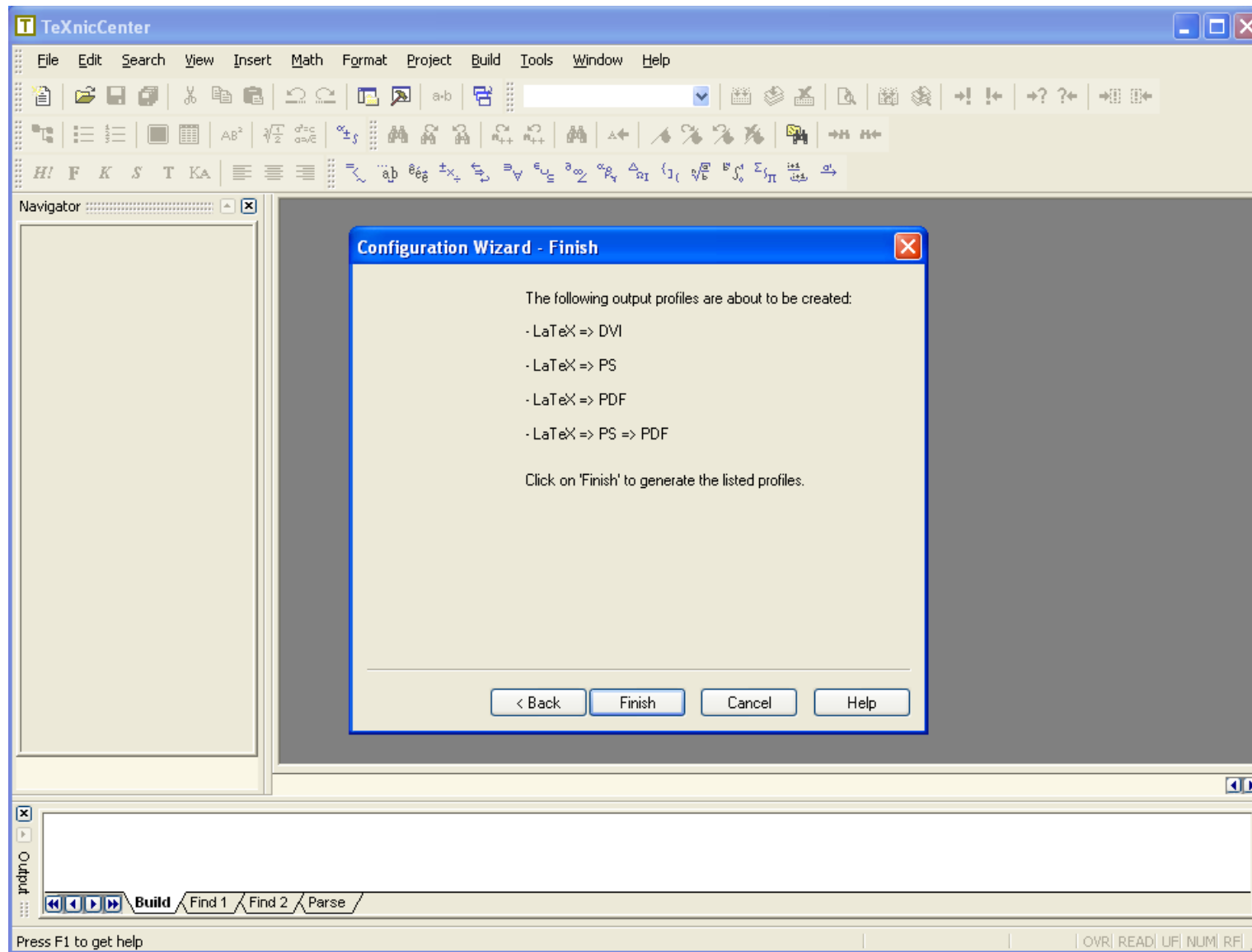


Append `\miktex\bin` to the MiKTeX install path.

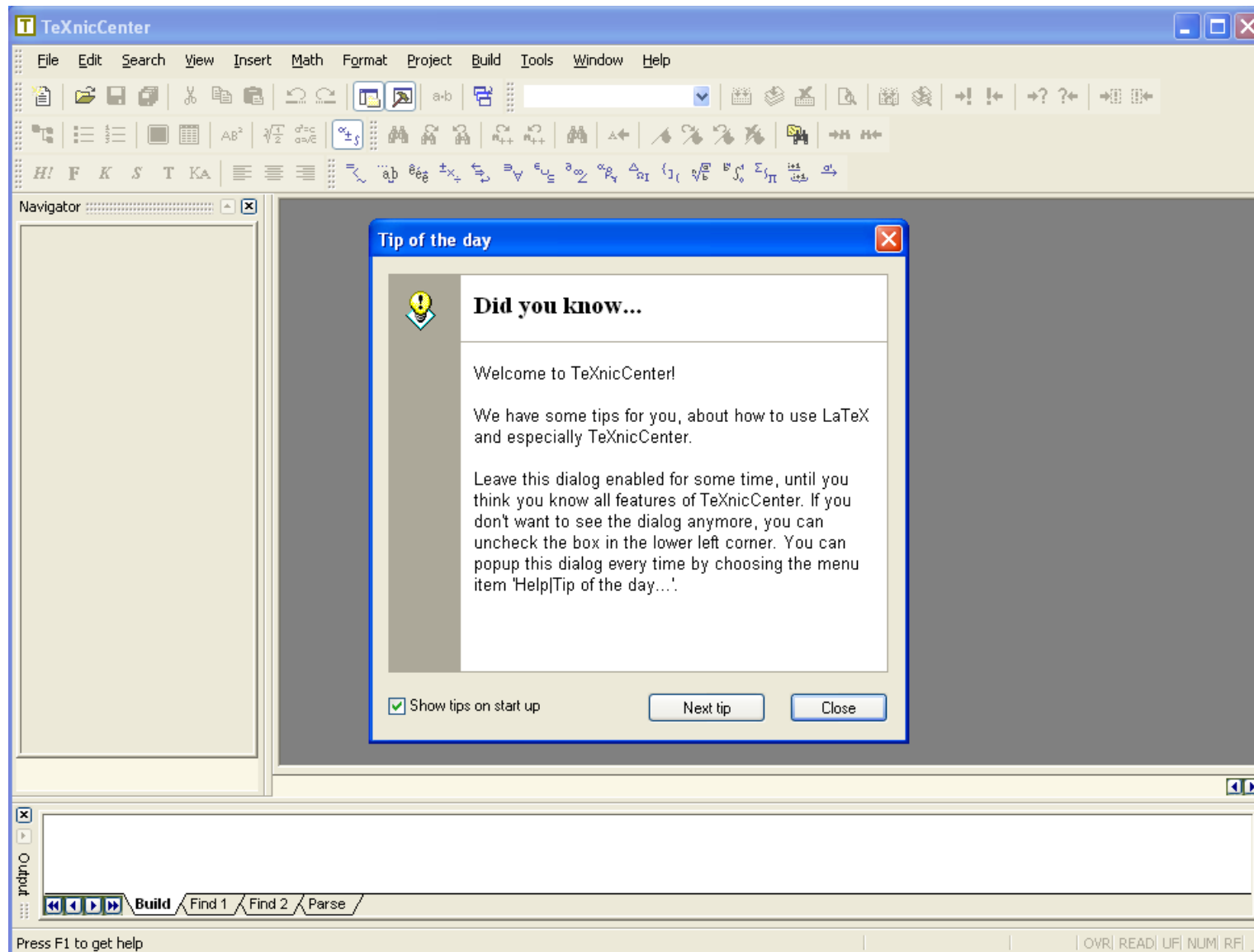
TeXnicCenter: Config Wizard



TeXnicCenter: Config Wizard



TeXnicCenter: Running



Basic L^AT_EX

A L^AT_EX source file consists of free format text interspersed with commands to the L^AT_EX formatting engine. (It's important to use a text editor — not Word — to edit these files.)

Except for approximately 10 characters with special meaning to L^AT_EX, the **printable** characters in the source file are copied to the output document.

The most important document structural component is the paragraph. Paragraphs are specified by inserting a blank line in the source file.

The formatting engine takes each paragraph in the input file, formats it nicely (for instance, by tweaking the space between words), and outputs it.

Normally, multiple spaces in the source file are equivalent to a single space, and multiple blank lines are equivalent to a single blank line.

Basic L^AT_EX Commands

L^AT_EX commands start with a single backslash (`\`) followed either by one or more letters or by a single non-letter.

A L^AT_EX document begins with the `\documentclass` command, which tells L^AT_EX the base document layout to use:

```
\documentclass{article}
```

Following the `\documentclass` command itself is a parameter enclosed in braces.

- If required, multiple parameters are separated by commas.
- If there are no parameters, the braces are optional.
 - If the braces are omitted, L^AT_EX discards any spaces following the command. If you want the space preserved, the braces are required.

Basic L^AT_EX Commands

Following the `\documentclass` command is the **preamble**, which basically contains additional instructions for the L^AT_EX system. The preamble cannot generate any output.

Following the preamble is the document body, which must be enclosed by the following commands:

```
\begin{document}  
\end{document}
```

This is an example of an **environment**. All environments are strictly nested: the end environment must always match exactly to the corresponding begin environment.

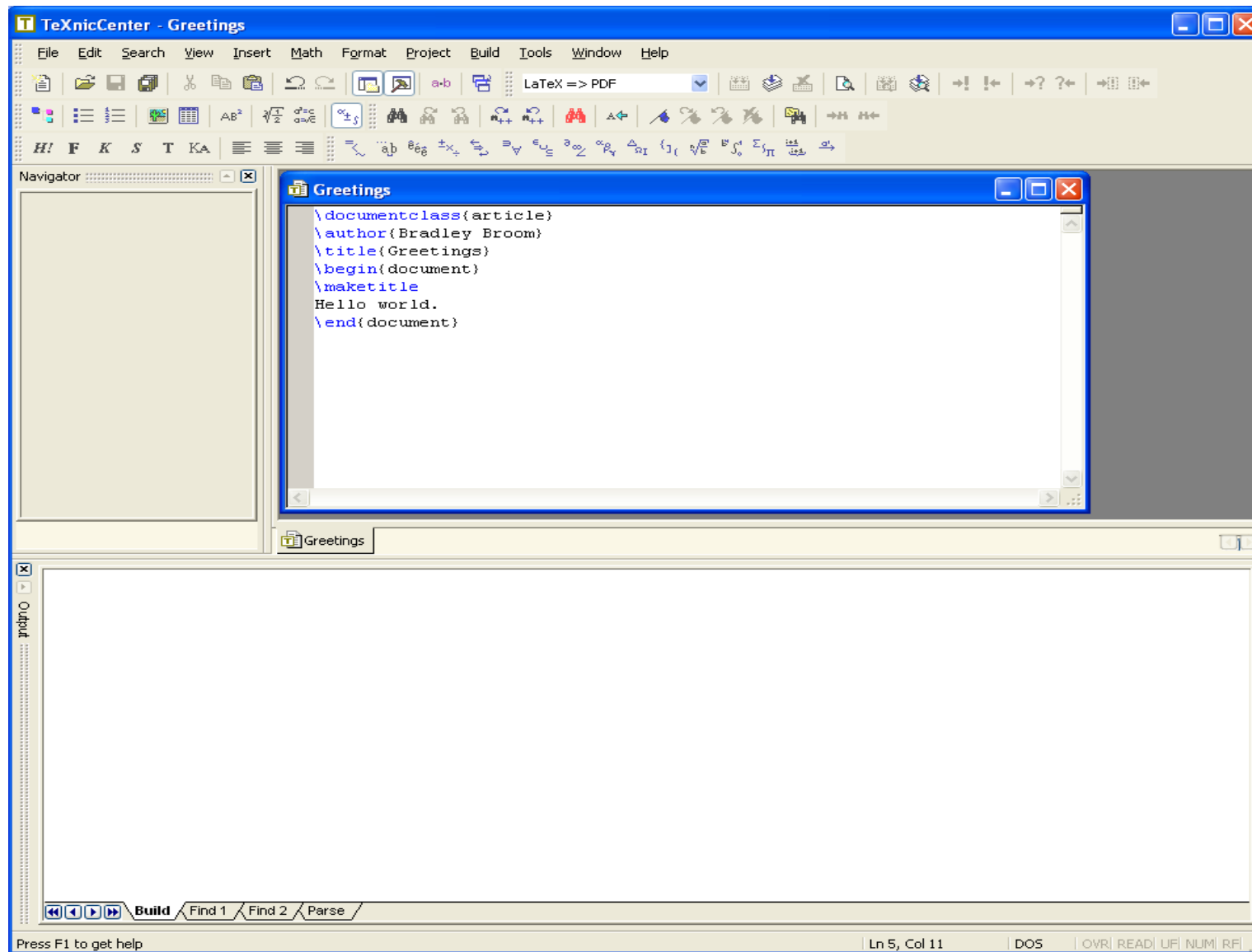
Anything following the `\end{document}` is ignored.

More Information

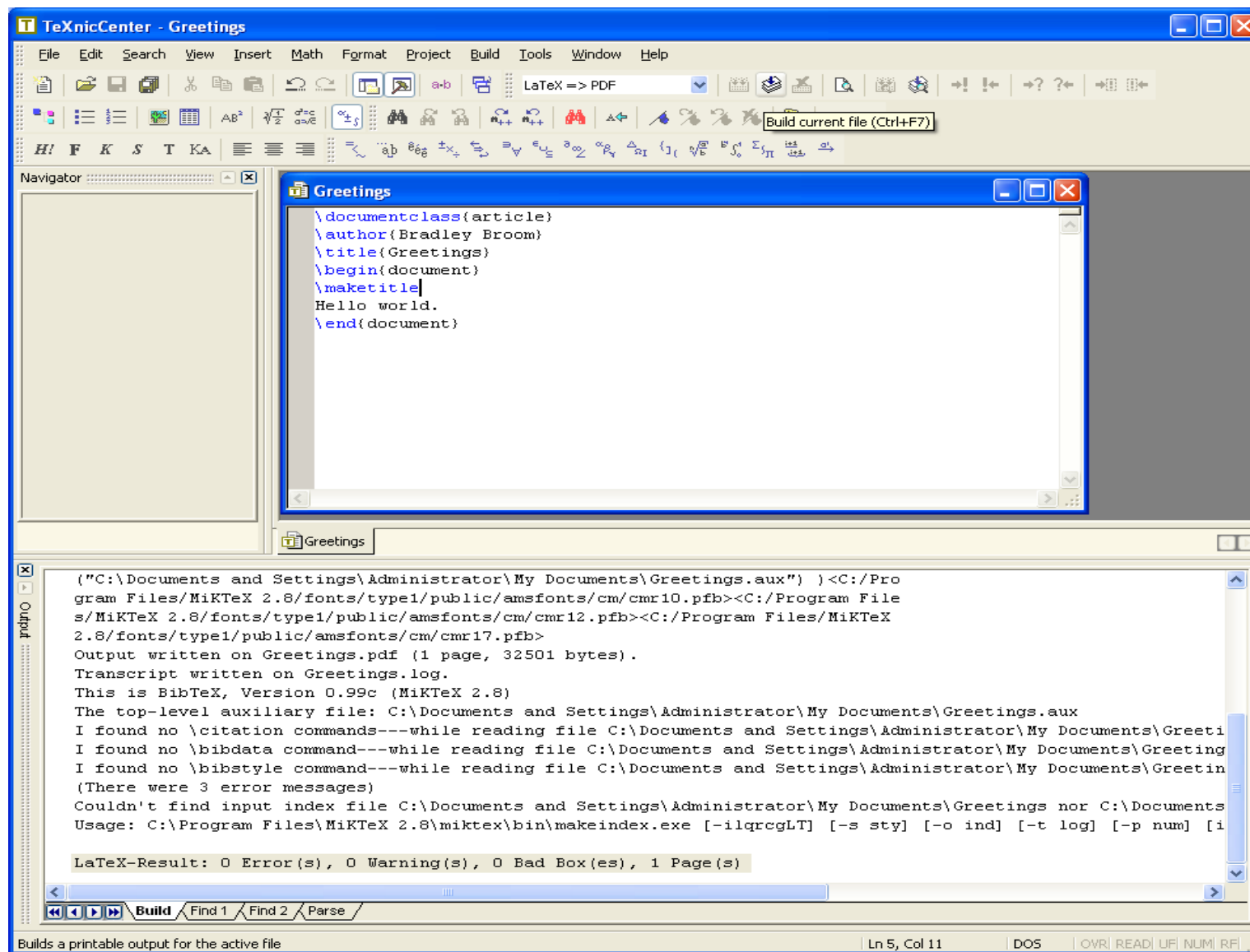
From the [CTAN Starting out](#) page, follow the links to:

- the *(Not So) Short Introduction to L^AT_EX 2_ε* ([lshort.pdf](#)) and read chapters 1, 2, and 4 (except section 4.1).
- the tutorials by Andrew Roberts and read tutorials 1 (ignoring the stuff about dvi output and converting to pdf — we will produce pdf directly) and 2.

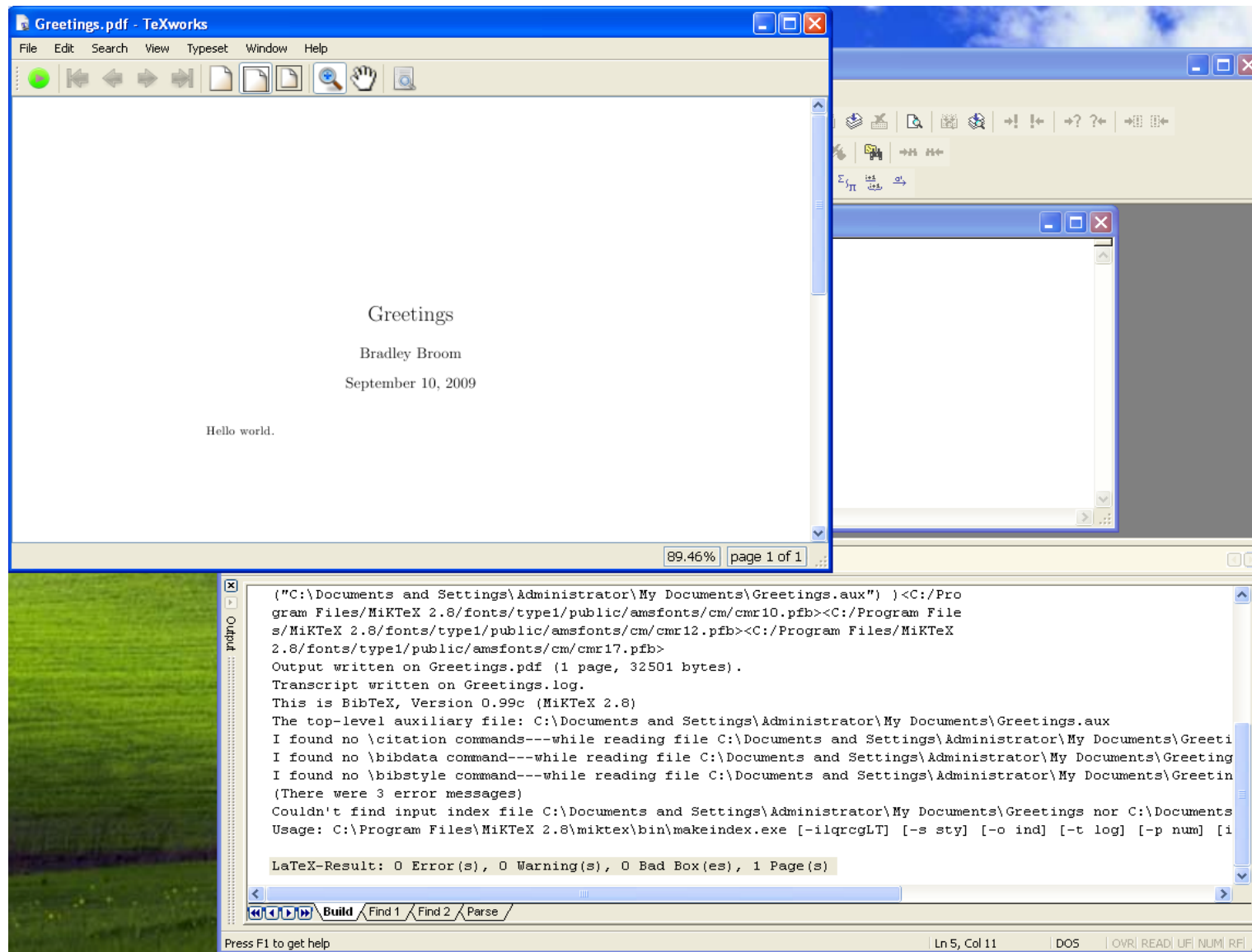
TeXnicCenter: Simple L^AT_EX Document



TeXnicCenter: Converting L^AT_EX to PDF



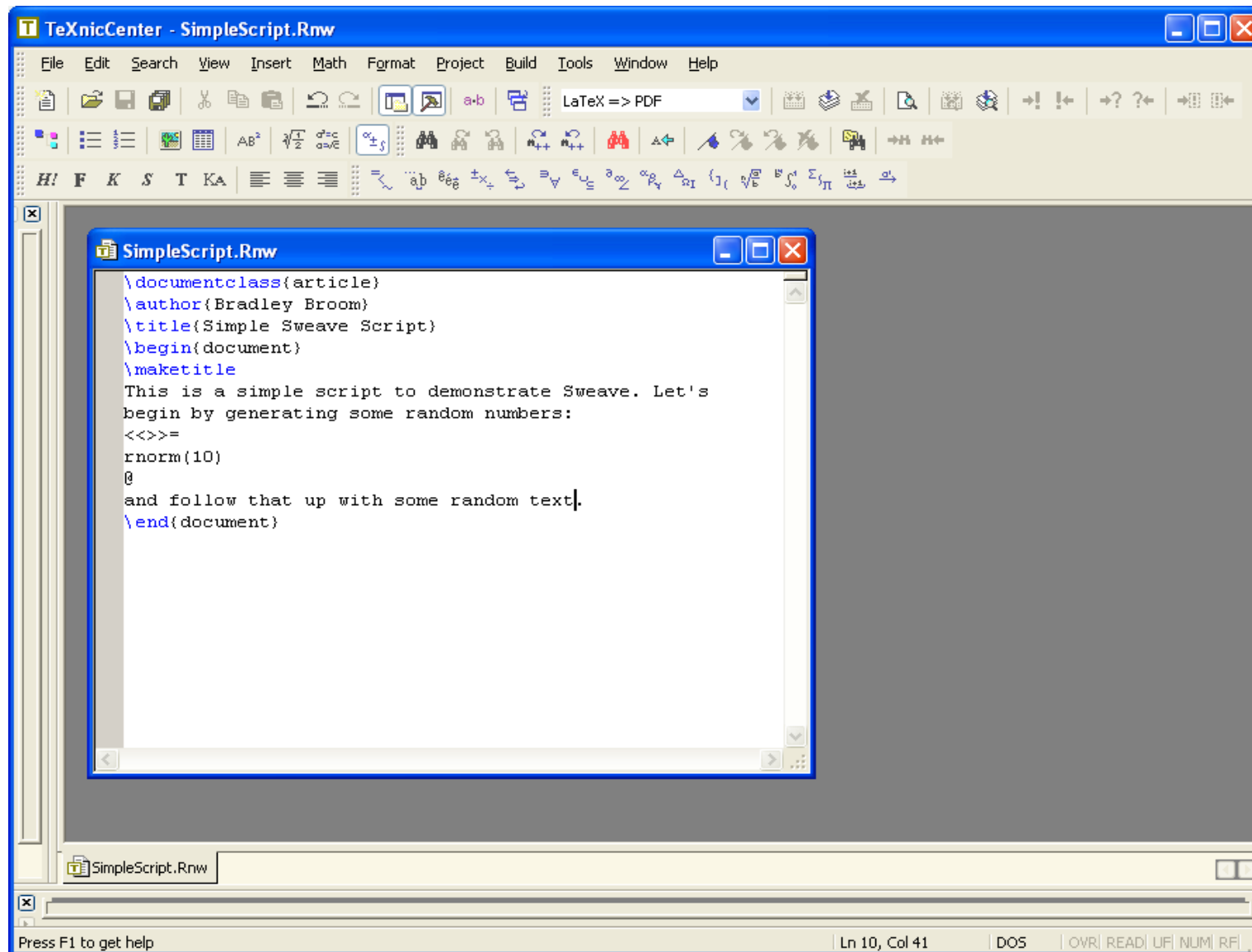
TeXnicCenter: Viewing PDF



Writing Documented R Analyses

1. Prepare a \LaTeX document describing the analysis. Give it an “Rnw” extension instead of “tex”. Say it is called “myfile.Rnw”
 - If you use TeXnicCenter, make sure it doesn’t silently append an invisible `.tex` extension.
2. Insert one or more R code chunks starting with `<<>>=`
3. Terminate each R code chunk with an “at” sign (`@`) followed by a space.

TeXnicCenter: Simple Sweave Script

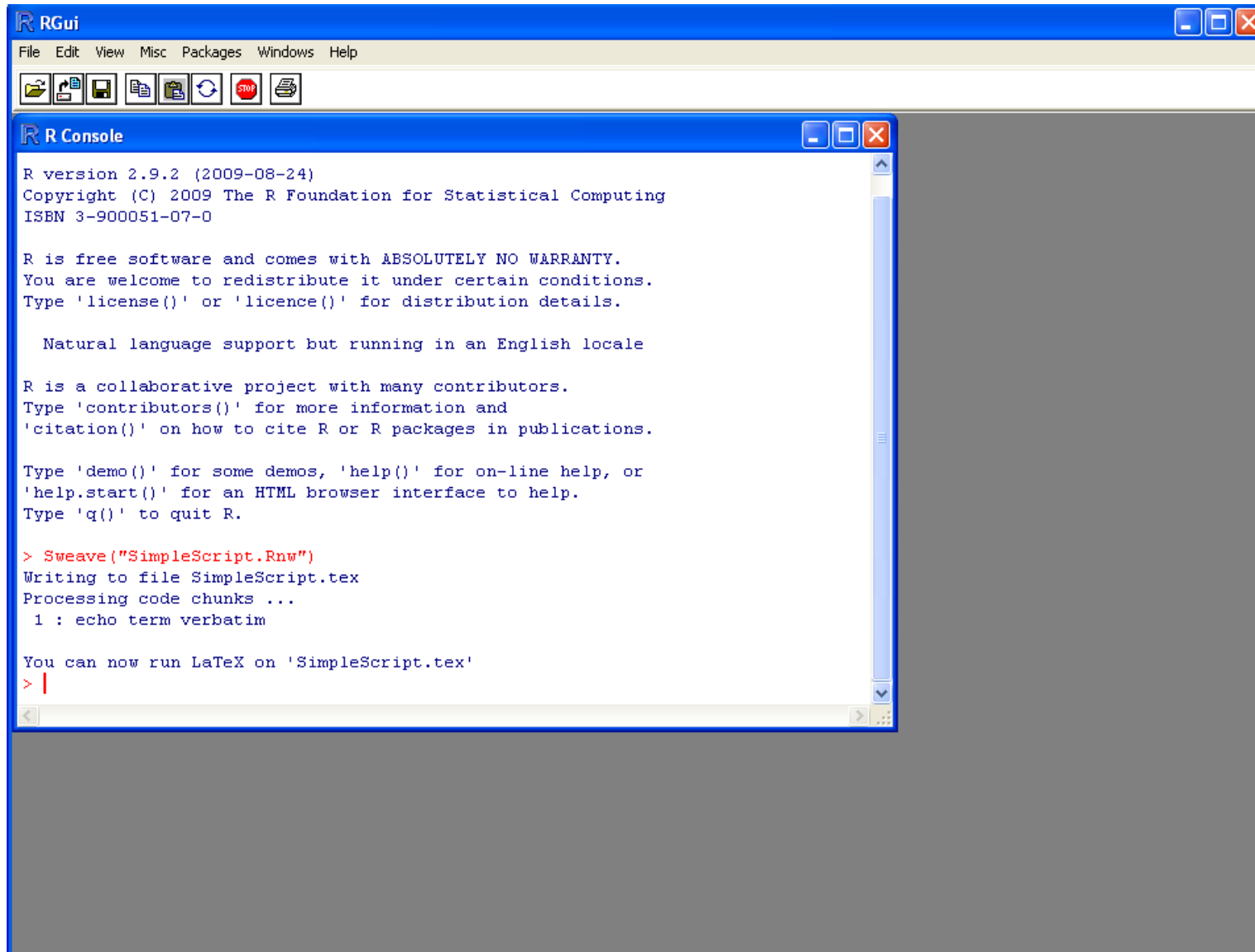


The screenshot shows the TeXnicCenter application window titled "TeXnicCenter - SimpleScript.Rnw". The interface includes a menu bar (File, Edit, Search, View, Insert, Math, Format, Project, Build, Tools, Window, Help), a toolbar with icons for file operations and LaTeX commands, and a main editing area. The editing area contains a window titled "SimpleScript.Rnw" with the following LaTeX Sweave script:

```
\documentclass{article}
\author{Bradley Broom}
\title{Simple Sweave Script}
\begin{document}
\maketitle
This is a simple script to demonstrate Sweave. Let's
begin by generating some random numbers:
<<>>=
rnorm(10)
@
and follow that up with some random text|.
\end{document}
```

The status bar at the bottom of the window displays "Press F1 to get help", "Ln 10, Col 41", "DOS", and "OVR| READ| U|F| NUM| RF| ..".

TeXnicCenter: Simple Sweave Script



```
RGui
File Edit View Misc Packages Windows Help
[Icons]

R Console
R version 2.9.2 (2009-08-24)
Copyright (C) 2009 The R Foundation for Statistical Computing
ISBN 3-900051-07-0

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> Sweave("SimpleScript.Rnw")
Writing to file SimpleScript.tex
Processing code chunks ...
 1 : echo term verbatim

You can now run LaTeX on 'SimpleScript.tex'
> |
```

TeXnicCenter: Simple Sweave Script

The screenshot shows two overlapping windows. The top window is RGui, and the bottom window is a Command Prompt.

RGui R Console:

```
R version 2.9.2 (2009-08-24)
Copyright (C) 2009 The R Foundation for Statistical Computing
ISBN 3-900051-07-0

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

Natural language support but running in an english locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> Sweave("SimpleScript.Rnw")
Writing to file SimpleScript.tex
Processing code chunks ...
  1 : echo term verbatim

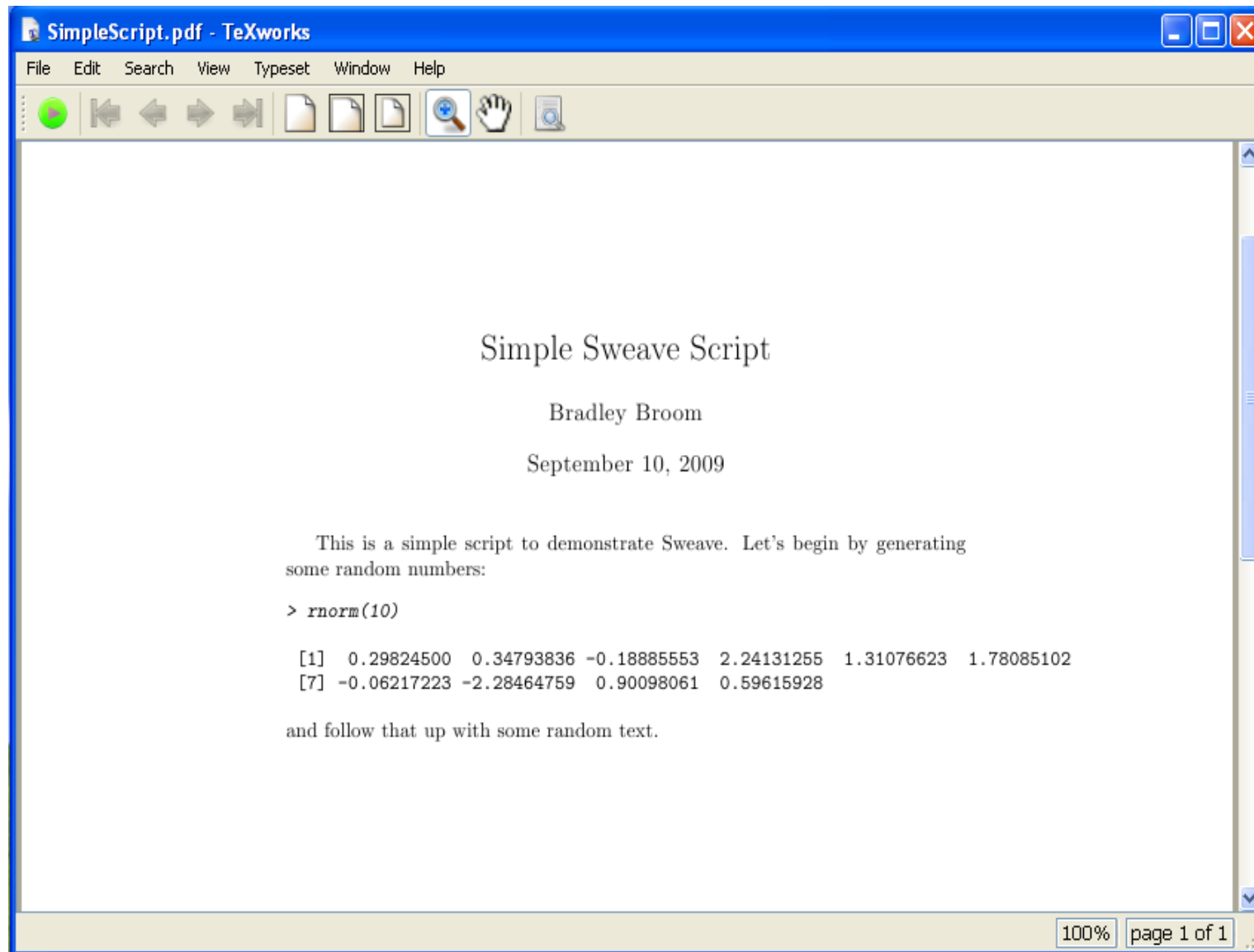
You can now run LaTeX on 'SimpleScript.tex'
> |
```

Command Prompt:

```
C:\Documents and Settings\Administrator\My Documents>pdflatex SimpleScript.tex
This is pdfTeX, Version 3.1415926-1.40.10 (MiKTeX 2.8)
entering extended mode
<"C:\Documents and Settings\Administrator\My Documents\SimpleScript.tex"
LaTeX2e <2005/12/01>
Babel <v3.81> and hyphenation patterns for english, dumylang, nohyphenation, ge
rman, ngerman, german-x-2009-06-19, ngerman-x-2009-06-19, french, loaded.
<"C:\Program Files\MiKTeX 2.8\tex\latex\base\article.cls"
Document Class: article 2005/09/16 v1.4f Standard LaTeX document class
<"C:\Program Files\MiKTeX 2.8\tex\latex\base\size10.clo">
<"C:\texmf\sweave\tex\latex\local\Sweave.sty"
<"C:\Program Files\MiKTeX 2.8\tex\latex\base\ifthen.sty">
<"C:\Program Files\MiKTeX 2.8\tex\latex\graphics\graphicx.sty"
<"C:\Program Files\MiKTeX 2.8\tex\latex\graphics\keyval.sty">
<"C:\Program Files\MiKTeX 2.8\tex\latex\graphics\graphics.sty"
<"C:\Program Files\MiKTeX 2.8\tex\latex\graphics\trig.sty">
<"C:\Program Files\MiKTeX 2.8\tex\latex\00miktex\graphics.cfg">
<"C:\Program Files\MiKTeX 2.8\tex\latex\pdfTeX-def\pdfTeX.def">>>
<"C:\Documents and Settings\Administrator\Application Data\MiKTeX\2.8\tex\latex
fancyvrb\fancyvrb.sty"
Style option: 'fancyvrb' v2.7a, with DG/SPQR fixes, and firstline=lastline fix
<2008/02/07> <tvz>> <"C:\texmf\sweave\tex\latex\local\upquote.sty"
<"C:\Program Files\MiKTeX 2.8\tex\latex\base\textcomp.sty">
<"C:\Program Files\MiKTeX 2.8\tex\latex\base\tslenc.def">>>
<"C:\Program Files\MiKTeX 2.8\tex\latex\base\fontenc.sty">
<"C:\Program Files\MiKTeX 2.8\tex\latex\base\tlenc.def">>
<"C:\Program Files\MiKTeX 2.8\tex\latex\ae\ae.sty"
<"C:\Program Files\MiKTeX 2.8\tex\latex\base\fontenc.sty">
<"C:\Program Files\MiKTeX 2.8\tex\latex\base\tlenc.def">
<"C:\Program Files\MiKTeX 2.8\tex\latex\ae\tlaer.fd">>>
No file SimpleScript.aux.
<"C:\Program Files\MiKTeX 2.8\tex\latex\base\ts1cmr.fd">
<"C:\Program Files\MiKTeX 2.8\tex\context\base\supp-pdf.tex"
Loading MPS to PDF converter (version 2006.09.02).]
> <"C:\Program Files\MiKTeX 2.8\tex\latex\ae\tlaett.fd"> l1<"C:\Documents and Se
ttings\Administrator\Local Settings\Application Data\MiKTeX\2.8\pdfTeX\config/p
dfTeX.map>]
<"C:\Documents and Settings\Administrator\My Documents\SimpleScript.aux"> ><"C:/
Program Files/MiKTeX 2.8/fonts/type1/public/amsfonts/cm/cmr10.pfb"><"C:/Program F
iles/MiKTeX 2.8/fonts/type1/public/amsfonts/cm/cmr12.pfb"><"C:/Program Files/MiKT
eX 2.8/fonts/type1/public/amsfonts/cm/cmr17.pfb"><"C:/Program Files/MiKTeX 2.8/fo
nts/type1/public/amsfonts/cm/cmsl10.pfb"><"C:/Program Files/MiKTeX 2.8/fonts/ty
pe1/public/amsfonts/cm/cmtt10.pfb">
Output written on SimpleScript.pdf (1 page, 57963 bytes).
Transcript written on SimpleScript.log.

C:\Documents and Settings\Administrator\My Documents>
```

TeXnicCenter: Simple Sweave Script



TeXnicCenter: Simple Sweave Script

The screenshot shows the TeXnicCenter interface with three windows open:

- SimpleScript.Rnw**: Contains the LaTeX preamble and the start of the document.


```

\documentclass{article}
\author{Bradley Broom}
\title{Simple Sweave Script}
\begin{document}
\maketitle
This is
random
<<>>=
rnorm(1
@
and fol
\end{do

```
- SimpleScript.tex**: Shows the Sweave script and its output.


```

\documentclass{article}
\author{Bradley Broom}
\title{Simple Sweave Script}
\usepackage{Sweave}
\begin{document}
\maketitle
This is a simple script to demonstrate Sweave. Let's begin by generating some
random numbers:
\begin{Schunk}
\begin{Sinput}
> rnorm(10)
\end{Sinput}
\begin{Soutput}
[1] 0.29824500 0.34793836 -0.18885553 2.24131255 1.31076623 1.78085102
[7] -0.06217223 -2.28464759 0.90098061 0.59615928
\end{Soutput}
\end{Schunk}

```
- Output**: Shows the compilation log.


```

The top-level auxiliary file: C:\Documents and Settings\Administrator\My Documents\SimpleScript.aux
I found no \citation commands---while reading file C:\Documents and Settings\Administrator\My Document
I found no \bibdata command---while reading file C:\Documents and Settings\Administrator\My Documents\
I found no \bibstyle command---while reading file C:\Documents and Settings\Administrator\My Documents
(There were 3 error messages)
Couldn't find input index file C:\Documents and Settings\Administrator\My Documents\SimpleScript nor C
Usage: C:\Program Files\MiKTeX 2.8\miktex\bin\makeindex.exe [-ilqrcgLT] [-s sty] [-o ind] [-t log] [-r
LaTeX-Result: 0 Error(s). 0 Warning(s). 0 Bad Box(es). 1 Page(s)

```

Make sure you never edit the .tex file: open it read-only.

Using Sweave

To produce the final document

1. In an R session, issue the command

```
Sweave("myfile.Rnw")
```

This executes the R code, inserts input commands and output computations and figures into a \LaTeX file called “myfile.tex”.

2. In the UNIX or DOS window (or using your favorite graphical interface), issue the command

```
pdflatex myfile
```

This produces a PDF file that you can use as you wish.

Viewing The Results

Here is a simple example, showing how the R input commands can generate output that is automatically included in the \LaTeX output of Sweave.

```
> x <- rnorm(30)
> y <- rnorm(30)
> mean(x)
```

```
[1] 0.2279967
```

```
> cor(x, y)
```

```
[1] 0.3408799
```

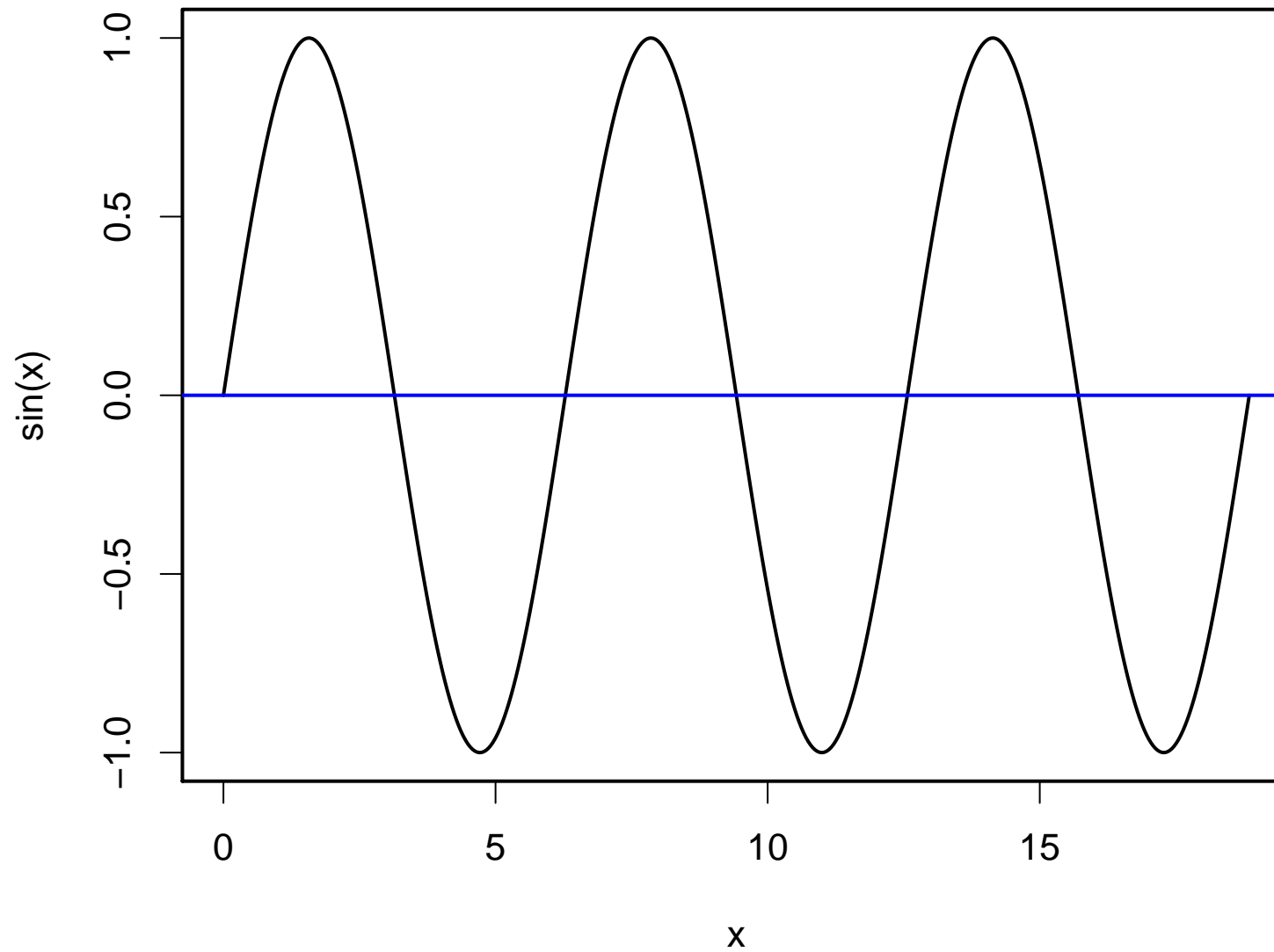
A Figure

Next, we are going to insert a figure. First, we can look at the R commands that are used to produce the figure.

```
> x <- seq(0, 6 * pi, length = 450)
> par(bg = "white", lwd = 2, cex = 1.3, mai = c(1.2,
+       1.2, 0.2, 0.2))
> plot(x, sin(x), type = "l")
> abline(h = 0, col = "blue")
```

On the next slide, we can look at the actual figure. (Part of the point of this example is to illustrate that you can separate the input from the output. You can even completely hide the input in the source file and just include the output in the report.)

Sine Curve



A Table

```
> library(xtable)
> x <- data.frame(matrix(rnorm(12), nrow = 3,
+   ncol = 4))
> dimnames(x) <- list(c("A", "B", "C"), c("C1",
+   "C2", "C3", "C4"))
> tab <- xtable(x, digits = c(0, 3, 3, 3, 3))
> tab
```

	C1	C2	C3	C4
A	1.052	-0.720	0.015	-0.595
B	0.159	-1.059	0.321	1.753
C	-0.539	0.530	-0.734	0.119

A Table, Repeated

Again, we want to point out that you can show the results—including tables—without showing the commands that generate them.

	C1	C2	C3	C4
A	1.052	-0.720	0.015	-0.595
B	0.159	-1.059	0.321	1.753
C	-0.539	0.530	-0.734	0.119

Beyond Matrices

We have gone from scalar to vector to matrix, attaching names as we go, with the goal of keeping associated information together. So far, we've done this with numbers, but we could use character strings instead:

```
> letters[1:3]
```

```
[1] "a" "b" "c"
```

```
> x <- letters[1]
```

```
> x <- letters[1:3]
```

```
> x <- matrix(letters[1:12], 3, 4)
```

Mixing Modes in Lists

In R, we cannot easily mix data of different modes in a vector or matrix:

```
> x <- c(1, "a")
```

```
> x
```

```
[1] "1" "a"
```

However, a list can have (named) components that are of different modes and even different sizes:

```
> x <- list(teacher = "Keith", n.students = 14,
```

```
+   grades = letters[c(1:4, 6)])
```

```
> x
```

```
$teacher
```

```
[1] "Keith"
```

```
$n.students
```

```
[1] 14
```

```
$grades
```

```
[1] "a" "b" "c" "d" "f"
```

Note that we named the components of the list at the same time that we created it. Many functions in R return answers as lists.

Extracting Items From Lists

If we want to access the first element of `x`, we might try using the index or the name in single brackets:

```
> x[1]
```

```
$teacher  
[1] "Keith"
```

```
> x["teacher"]
```

```
$teacher  
[1] "Keith"
```

These don't quite work. The single bracket extracts a component, but

keeps the same mode; what we have here is a list of length 1 as opposed to a character string. Two brackets, on the other hand...

```
> x[[1]]
```

```
[1] "Keith"
```

```
> x[["teacher"]]
```

```
[1] "Keith"
```

The double bracket notation can be cumbersome, so there is a shorthand notation with the dollar sign. Using names keeps the goals clear.

```
> x$teacher
```

```
[1] "Keith"
```

Lists with Structure

Now, there are some very common types of structured arrays. The most common is simply a table, where the rows correspond to individuals and the columns correspond to various types of information (potentially of multiple modes). Because we want to allow for multiple modes, we can construct a table as a list, but this list has a constraint imposed on it – all of its components must be of the same length. This is similar in structure to the idea of a matrix that allows for multiple modes. This structure is built into R as a `data frame`.

This structure is important for data import. Before looking at that, however, we are going to revisit the notion of reproducibility of our analyses.

Reading Data Into R

While we can simply type stuff in, or use `source()` to pull in small amounts of data we've typed into a file, what we often want to do is to read a big table of data. R has several functions that allow us to do this, including `read.table()`, `read.delim()`, and `scan()`.

We can experiment by using some of the files that we generated in dChip for the first HWK.

We could load the sample info file, and the list of filtered genes. Then we could use the sample info values to suggest how to contrast the expression values in the filtered gene table.

Importing our dChip Data

I exported all of the dChip quantifications to a single file. The file has a header row, with columns labeled “probe set”, “gene”, “Accession”, “LocusLink”, “Description” and then “N01” and so on, 1 column per sample. We can read this into R as follows:

```
> singh.dchip.data <-  
  read.delim(c("../SinghProstate/Singh_",  
              "Prostate_dchip_expression.xls"));  
> class(singh.dchip.data)  
[1] "data.frame"  
> dim(singh.dchip.data)  
[1] 12625  108
```

The number of columns is a bit odd...

More on Importing

If we invoke `help(read.delim)`, help pops up for `read.table`. The former is a special case of the latter. Let's take a look at bits of the usage lines for each:

```
read.table(file, header = FALSE, sep = "",
  quote = "\"", dec = ".", row.names, col.names,
  as.is = FALSE, na.strings = "NA", colClasses = NA,
  nrow = -1, skip = 0, check.names = TRUE,
  fill = !blank.lines.skip, strip.white = FALSE,
  blank.lines.skip = TRUE, comment.char = "#")
```

```
read.delim(file, header = TRUE, sep = "\t", quote=
  "\"", dec=".", fill = TRUE, ...)
```

Note the default function arguments!

Speeding Up Import

Reading the documentation suggests a few speedups:

- we can use `comment.char = ""`, speeding things up
- we can use `nrows = 12626`, for better memory usage
- we could shift to using `scan` (use help!).

```
singh.dchip.data <-  
  read.delim(c("../SinghProstate/Singh_Prostate"  
              , "_dchip_expression.xls"),  
            comment.char = "",  
            nrows = 12626  
            );
```

is indeed faster!

Is This What We Want?

All of the expression data is now nicely loaded in a data frame. But this data frame really breaks into two parts quite nicely – gene information, and expression values. If we split these apart, then the expression value matrix has 102 columns, corresponding to the sample info entries quite nicely.

```
singh.annotation <- singh.dchip.data[,1:5];  
singh.dchip.expression <-  
  as.matrix(singh.dchip.data[,6:107]);  
rownames(singh.dchip.expression) <-  
  singh.annotation$probe.set;
```

Grab the Sample Info Too

What are the columns in my sample info file?

```
scan name          sample name      type
  run_date_block  cluster_block
N01__normal       N01          N          2          2
```

(the last two you might not have).

```
singh.sample.info <-
  read.delim("../SinghProstate/sample_info_2.txt",
             comment.char = "",
             nrow = 103
  );
```

Test Something Interesting

In the first homework, we saw that the data split into two clusters that didn't agree well with the tumor/normal split. It might very well be that there was some type of batch effect in addition to the biological split of interest.

Can we factor the batch effect out? If we know what the batch split is, we can first fit a model using just the batches, subtract the fit off, and then fit a model using the tumor/normal split on what remains.

Tumor vs Normal

```
singh.probeset.lm <-  
  lm(singh.dchip.expression[  
    singh.annotation$probe.set  
    == "31539_r_at",]  
    ~ singh.sample.info$type  
  );  
singh.probeset.anova <-  
  anova(singh.probeset.lm);
```


Tumor vs Normal (cont)

```
> singh.probeset.anova
```

```
Analysis of Variance Table
```

```
Response: singh.dchip.expression[
```

```
  singh.annotation$probe.set == "31539_r_at",]
```

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
\$type	1	71.42	71.42	5.3748	0.02247 *
Residuals	100	1328.81	13.29		

```
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

T vs N, After Blocking

```
singh.probeset.lm.full <-  
  lm(singh.dchip.expression[  
    singh.annotation$probe.set  
    == "31539_r_at",]  
    ~ singh.sample.info$cluster.block  
    + singh.sample.info$type  
  );  
singh.probeset.anova.full <-  
  anova(singh.probeset.lm.full);
```

T vs N, After Blocking (cont)

```
> singh.probeset.anova.full
```

```
Analysis of Variance Table
```

```
Response: singh.dchip.expression[
```

```
  singh.annotation$probe.set == "31539_r_at",]
```

	Df	Sum Sq	Mean Sq	F value	Pr(>F)	
\$block	1	404.97	404.97	40.6399	5.85e-09	***
\$type	1	8.75	8.75	0.8779	0.3511	
Residuals	99	986.51	9.96			

```
---
```

```
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Hasn't Someone Done This?

Other people have thought about the data structures that might be natural for microarray data. In particular, a lot of these functions are collected at Bioconductor.

Let's try to grab some of the packages and functions that will help with this type of analysis.

Obtaining extra R packages

The R GUI makes it easy to get additional packages via the internet. From the “Packages” menu, you simply select “Install package(s)...”. (In order to install packages from Bioconductor, you must first use the “Select repositories...” menu item to tell R to look there.) The menu item presents a dialog box containing a list of the available packages. You then select one or more (by holding the control key while clicking with the mouse) and press the “OK” button. R then downloads the package, installs it, and updates the help files. It finishes by asking if you want to delete the downloaded files; unless you want to save them to install them on another computer without an internet connection, the usual answer is “yes”.

Bioconductor Packages

You will need the following packages from the [Bioconductor web site](#). Use the items “Select repositories...” and “Install package(s)...” on the “Packages” menu to get them.

reposTools : Repository tools for R

Biobase : Base functions for BioConductor

affy : Methods for Affymetrix oligonucleotide arrays

affydata : Affymetrix data for demonstration purposes

affypdnn : Probe dependent nearest neighbor (PDNN) for the affy package

Bioconductor Widget Packages

In order to use some of the graphical tools that make it easier to read Affymetrix microarray data and construct sensible objects describing the experiments, you will also need the following packages from the [Bioconductor web site](#).

tkWidgets : R based Tk widgets

widgetTools : Creates an interactive tcltk widget

DynDoc : Dynamic document tools