SAMPLE THESIS CREATED BY USING LyX

By
Ahmed Mohamed Rashed Desoki

A Thesis Submitted to the
Faculty of Engineering at Cairo University
in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE
in
Aerospace Engineering

FACULTY OF ENGINEERING, CAIRO UNIVERSITY
GIZA, EGYPT
August, 2020

Proudly created by

Except for the figures created by Matlab[1], this thesis has been created by *open source software* (OSS) packages. Special thanks go to the numerous generous developers behind the following projects:

GNU project free software, mass collaboration project aiming to give users freedom

LaTeX document markup language

TeX Live cross-platform LaTeX distribution

MiKTeX LaTeX distribution for Windows

LyX cross-platform LaTeX-based document preparation system

Beamer LaTeX class for creating presentation slides and handouts

Inkscape cross-platform vector graphics editor

TeX Text Inkscape plugin for creating and editing LaTeX formulae

Other great projects I failed to mention ...

Other software packages

Other software packages that greatly helped me during this research include:

Areca cross-platform incremental backup package

pdfcrop a Perl program for removing white margins of a pdf file; indispensable for exported Matlab figures

GoldenDict cross-platform feature-rich dictionary lookup program

[1] For your information, NumPy + SciPi + Matplotlib + Spyder offer very competitive alternative to Matlab. For Windows, all these packages and more are distributed by *Python(x,y)*.

SAMPLE THESIS CREATED BY USING LyX

By
Ahmed Mohamed Rashed Desoki

A Thesis Submitted to the
Faculty of Engineering at Cairo University
in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE
in
Aerospace Engineering

Under the Supervision of

Prof. Name1 Name1 Name1

Professor
Aerospace Engineering Department
Faculty of Engineering, Cairo University

Prof. Name2 Name2 Name2

Associate Professor
Aerospace Engineering Department
Faculty of Engineering, Cairo University

Prof. Name3 Name3 Name3

Assistant Professor
Aerospace Engineering Department
Faculty of Engineering, Cairo University

FACULTY OF ENGINEERING, CAIRO UNIVERSITY
GIZA, EGYPT
August, 2020

This page is intentionally left blank!

SAMPLE THESIS CREATED BY USING LyX

By
Ahmed Mohamed Rashed Desoki

A Thesis Submitted to the
Faculty of Engineering at Cairo University
in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE
in
Aerospace Engineering

Approved by the Examining Committee

Prof. Name1 Name1 Name1, thesis main advisor

Associate Prof. Name3 Name3 Name3, internal examiner

Prof. Name4 Name4 Name4, external examiner, National Research Center

FACULTY OF ENGINEERING, CAIRO UNIVERSITY
GIZA, EGYPT
August, 2020

This page is intentionally left blank!

Engineer's Name: Ahmed Mohamed Rashed Desoki
Date of Birth 1 / 1 / 1980
Nationality: Egyptian
E-mail: email@yahoo.com
Phone: 01223456789
Address: address address address

Registration Date: 1 / 1 / 2015
Awarding Date: / / 2018
Degree: Master of Science
Department: Aerospace Engineering
Supervisors:

Prof. Name1 Name1 Name1
Dr. Name2 Name2 Name2

Examiners:

Prof. Name1 Name1 Name1, thesis main advisor
Associate Prof. Name3 Name3 Name3, internal examiner
Prof. Name4 Name4 Name4, external examiner

Title of Thesis:

SAMPLE THESIS CREATED BY USING LyX

Key Words:

Keyword1; Keyword2; Keyword3; Keyword4; Keyword5; Keyword6; Keyword7; Keyword8; Keyword9; Keyword9; Keyword10.

Summary:

This abstract is incomplete. Refer to the complete abstract on page i.

Word word word

This page is intentionally left blank!

Abstract

I'm Ahmed Mohamed Rashed Desoki, an assistant professor at Aerospace Engineering Department, Cairo University.

I created this thesis template to show you how you can create a professional thesis using Open Source Software (OSS). Chapters of this template themselves concisely explain the necessary background you need to know about LaTeX, LyX, floating figures and tables, equations, references management, vector graphics, Inkscape, including program codes and others.

I strongly urge you to prepare your thesis file from the very beginning of your research. This is invaluable since it enables you to immediately document and cite every piece of new information you learn. I strongly urge you to stick to immediate documentation and citation as you learn. Citation itself expresses the value of your writing. You will find your citations invaluable especially after you read and learn a lot. At this time you will really fail to remember from where you learned every information.

This template is hosted at https://github.com/ahmed-rashed/ThesisTemplate. Usage of this template is licensed under GNU GPLv3[1]. If you just want to use this template, simply download it as a zip file using https://github.com/ahmed-rashed/ThesisTemplate/archive/master.zip and proceed. While you are using this template, if you faced problems, try hard to read, learn and dig for solutions by yourself. If you improved/corrected/debugged/extended this template, then please *clone* the template repository using **Git** by `$ git clone https://github.com/ahmed-rashed/ThesisTemplate.git`, and kindly[2] send me your modifications as a *pull request* . If you don't know what is **Git**, you can find concise explanation in chapter 10, or in [1].

Finally, foreign languages usually causes some problems to LaTeX documents. Arabic is not an exception. So if you faced a strange problem that you cannot solve, try disabling the Arabic parts of this thesis to check if the problem is related to the Arabic language[3]. To do so, just use the **Thesis_English.lyx** file. If disabling Arabic solved your problem, please try hard to find a solution and reactivate the Arabic again. **Arabic scientists cannot help their nations using any language other than Arabic.**

[1] www.gnu.org/licenses/quick-guide-gplv3.en.html

[2] In fact, you have to share your improvements according to the GNU GPLv3 license.

[3] Mostly the problem is not specific to Arabic, but to several other languages as well.

This page is intentionally left blank!

Acknowledgments

Thanks to the Allah who helped me completing this template. I ask him to accept it from me for the sake of his mercy.

This page is intentionally left blank!

Table of Contents

Abstract i

Acknowledgments iii

Table of Contents v

List of Tables vii

List of Figures ix

List of Codes xi

Nomenclature xiii

1 Word Processors; LaTeX vs MS Word 1

2 LaTeX; a Document Markup Language 3
2.1 LaTeX Integrated Development Environment (IDE) . . . 3
2.2 Porting a LaTeX Document . . . 3
2.3 Arabic Support . . . 6
2.4 Installing LaTeX . . . 6

3 LyX; a Graphical Front-End to LaTeX 7
3.1 Installing LyX . . . 7
3.2 Learning LyX . . . 8
3.3 Porting a LyX Document . . . 8
3.4 Arabic Support . . . 8

4 Floats, Figures, Tables and Equations 11
4.1 Concept of Floating Graphics, Tables . . . 11
4.2 Compound Figures . . . 11
4.2.1 Subfigure and Subtable . . . 11
4.3 Continued Floats . . . 11
4.4 Landscape Floats . . . 11
4.5 Side-by-Side Facing Floats . . . 11
4.6 Free Inline Graphics without Captions . . . 11
4.7 Tables . . . 12
4.8 Equations . . . 12
4.8.1 SDOF Mass Spring System . . . 12
4.8.2 Inverse Laplace Transform Derivation . . . 15

5 Reference Management Software **19**

6 Vector Graphics **21**
6.1 Raster vs Vector Graphics . . . 21
6.2 Vector Graphics Editors . . . 21

7 Inkscape; Free and Open Source Vector Graphics Editor **25**
7.1 Import Graphics from pdf . . . 25
7.2 Interesting Preferences . . . 26
7.3 Interesting Plug-ins . . . 27
7.3.1 Function Plotter . . . 27
7.3.2 TexText . . . 27
7.4 Learning Inkscape . . . 27

8 Including Program Codes **31**

9 About the Nomenclature **33**
9.1 Problems with Arabic . . . 33

10 Version Control Using Git **35**
10.1 Centralized vs Decentralized Version Control . . . 35
10.2 Introducing Git . . . 35
10.2.1 Git is Very Different . . . 35
10.2.2 Git GUI's . . . 36
10.2.2.1 Tower . . . 36
10.2.2.2 GitKraken . . . 37
10.2.3 Installing Git . . . 37
10.3 Workflow of Git . . . 37
10.3.1 Git Cheat Sheet . . . 37
10.3.2 Git Best Practices . . . 38
10.4 Git Terminology Explained . . . 38
10.5 Undoing Things . . . 49
10.5.1 Revert vs reset . . . 49
10.6 Merge Conflicts . . . 49
10.7 Diff of Complex Text Files of non-text files . . . 50
10.8 Further Details . . . 50
10.8.1 Excluding Files from Version Control . . . 50
10.8.2 Submodules . . . 50
10.8.3 Undoing Things . . . 50
10.8.4 Restore a Previous Version . . . 50

A Matlab Codes **51**

References **57**

Index **59**

List of Tables

1.1 LaTeX vs Microsoft Word 2

4.1 Table caption 13
4.2 Comparison between somethings 14

10.1 Typical centralized and decentralized VCS's. 36
10.2 Centralized versus decentralized VCS 37

This page is intentionally left blank!

List of Figures

1.1 Effort and time consumption of MS Word as compared to LaTeX. 2

2.1 LaTeX cheat sheet 4

3.1 Correcting svg converters in Inkscape 9

4.1 Figure composed of a subfigure and subtable 12
4.2 SDOF Mass Spring System 13

6.1 Sample raster graphics. This figure is forced to be on a left page for easier comparison with figure 6.2 on the opposite page. 22
6.2 Vector graphics version of figure 6.1 23

7.1 Vector graphic imported from the user guide of a home use ADSL router . 26
7.2 The Function Plotter plugin 28
7.3 Figure illustrating the capabilities of "Function Plotter" and "TexText" plug ins. 29

10.1 VCS illustration [1] 36
10.2 Git commands versus SVN commands [1] 38
10.3 Result of a survey about favorite VCS's 39
10.4 Git Basics [https://www.git-tower.com/learn/cheat-sheets/vcs-workflow with modifications] 40
10.5 Git branching and merging [https://www.git-tower.com/learn/cheat-sheets/vcs-workflow with modifications] 41
10.6 Git sharing work via *remote* repositories [https://www.git-tower.com/learn/cheat-sheets/vcs-workflow with modifications] 42
10.7 Git Cheat Sheet [https://www.git-tower.com/learn/cheat-sheets/git] 43
10.8 Git Best practices [https://www.git-tower.com/learn/cheat-sheets/git] 44
10.9 Basic *remote* workflow [1] 45
10.10 *merge* versus *rebase* 47
10.11 *fast forward merge* 48

This page is intentionally left blank!

List of Codes

A.1 SDOF_Free_Response_Visc_main 51
A.2 function SDOF_Free_Response_Visc.m 52
A.3 function export_figure 52

This page is intentionally left blank!

Nomenclature

DAG	Directed acyclic graph
GUI	Graphical User Interface
IDE	Integrated Development Environment
IRF	Impulse Response Function
MS	Microsoft
ode	ordinary differential equation
OSS	Open Source Software
PR	Pull Request
RCS	Revision Control System
SCM	Source Code Management
SDOF	Single Degree Of Freedom
SHA-1	Secure Hash Algorithm 1
TF	Transfer Function
VCS	Vevision Control System

This page is intentionally left blank!

Chapter 1

Word Processors; LaTeX vs MS Word

Usually there are two categories of word processing software packages; table 1.1

- What You See Is What You Get (WYSIWYG)
- What You See Is What You Mean (WYSIWYM)

Roughly, you can compare LaTeX to Word as you compare Matlab to Excel. Figure 1.1 visualizes the effort and time consumption needed.

By the way, if you are annoyed by the existence of table 1.1 and figure 1.1 at the following page, this is explained in http://tex.stackexchange.com/questions/66293/strange-behaviour-with-figure-on-chapter-first-page

WYSIWYG	WYSIWYM
Microsoft Word LibreOffice Writer AbiWord Calligra Words	LaTeX LyX

Table 1.1: LaTeX vs Microsoft Word

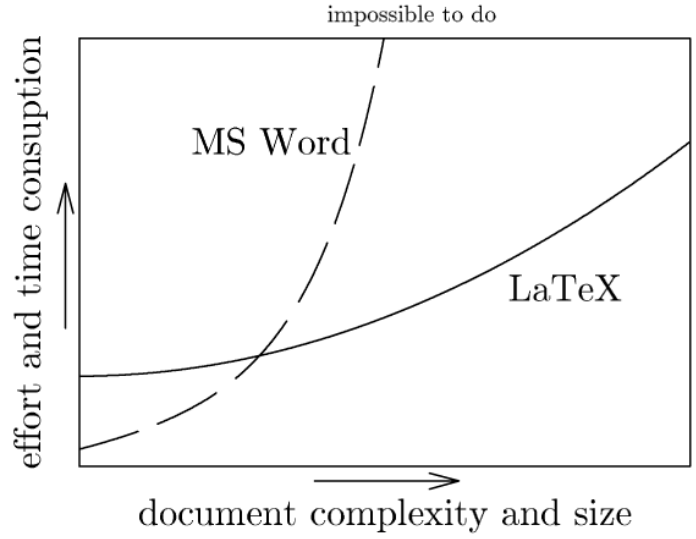


Figure 1.1: Effort and time consumption of MS Word as compared to LaTeX.

Chapter 2

LaTeX; a Document Markup Language

LaTeX is a document markup language.

- Simply you can think of it as similar to HTML[1]
- In order to create a document in LaTeX, a **.tex** file must be created using some **text editor**
- The **.tex** file is then **compiled** to produce the document
- LaTeX can generate several document formats including "pdf"

LaTeX is Free

Although being free is an advantage, but it is a drawback at the same time! Free implies:

- Slow download server
- No clean official documentation
- Several alternatives to do the same thing

However; LaTeX is very mature and widely used by professional/enterprise publishers

- Also it has a big user community
  - when you encounter a problem, google it. Most likely you will find others had encountered it and found a solution

2.1 LaTeX Integrated Development Environment (IDE)

To write a document using LaTeX, any text editor can be used.

- But using a dedicated LaTeX editor/IDE can greatly ease your job
- A dedicated LaTeX IDE:
  - can highlight and auto complete LaTeX keywords
  - has several LaTeX templates for several types of documents
  - facilitates compiling, solving compile errors and debugging
  - ...
- Sample LaTeX IDE's are Texstudio, Kile, ...

2.2 Porting a LaTeX Document

Usually LaTeX source files reference images and other external files. Hence, if you want to move/copy your LaTeX document to another computer, you have to move/copy all the

[1] (HyperText Markup Language)

LaTeX 2_ε Cheat Sheet

Document classes

`book`	Default is two-sided.
`report`	No `\part` divisions.
`article`	No `\part` or `\chapter` divisions.
`letter`	Letter (?).
`slides`	Large sans-serif font.

Used at the very beginning of a document: `\documentclass{`*class*`}`. Use `\begin{document}` to start contents and `\end{document}` to end the document.

Common `documentclass` options

`10pt/11pt/12pt`	Font size.
`letterpaper/a4paper`	Paper size.
`twocolumn`	Use two columns.
`twoside`	Set margins for two-sided.
`landscape`	Landscape orientation. Must use `dvips -t landscape`.
`draft`	Double-space lines.

Usage: `\documentclass[`*opt,opt*`]{`*class*`}`.

Packages

`fullpage` Use 1 inch margins.
`anysize` Set margins: `\marginsize{`*l*`}{`*r*`}{`*t*`}{`*b*`}`.
`multicol` Use *n* columns: `\begin{multicols}{`*n*`}`.
`latexsym` Use LaTeX symbol font.
`graphicx` Show image: `\includegraphics[width=`*x*`]{`*file*`}`.
`url` Insert URL: `\url{`*http://...*`}`.

Use before `\begin{document}`. Usage: `\usepackage{`*package*`}`

Title

`\author{`*text*`}`	Author of document.
`\title{`*text*`}`	Title of document.
`\date{`*text*`}`	Date.

These commands go before `\begin{document}`. The declaration `\maketitle` goes at the top of the document.

Miscellaneous

`\pagestyle{empty}`	Empty header, footer and no page numbers.
`\tableofcontents`	Add a table of contents here.

Document structure

`\part{`*title*`}`
`\chapter{`*title*`}`
`\section{`*title*`}`
`\subsection{`*title*`}`
`\subsubsection{`*title*`}`
`\paragraph{`*title*`}`
`\subparagraph{`*title*`}`

Use `\setcounter{secnumdepth}{`*x*`}` suppresses heading numbers of depth $> x$, where `chapter` has depth 0. Use a `*`, as in `\section*{`*title*`}`, to not number a particular item—these items will also not appear in the table of contents.

Text environments

`\begin{comment}`	Comment (not printed). Requires `verbatim` package.
`\begin{quote}`	Indented quotation block.
`\begin{quotation}`	Like `quote` with indented paragraphs.
`\begin{verse}`	Quotation block for verse.

Lists

`\begin{enumerate}`	Numbered list.
`\begin{itemize}`	Bulleted list.
`\begin{description}`	Description list.
`\item` *text*	Add an item.
`\item[`*x*`]` *text*	Use *x* instead of normal bullet or number. Required for descriptions.

References

`\label{`*marker*`}`	Set a marker for cross-reference, often of the form `\label{sec:item}`.
`\ref{`*marker*`}`	Give section/body number of marker.
`\pageref{`*marker*`}`	Give page number of marker.
`\footnote{`*text*`}`	Print footnote at bottom of page.

Floating bodies

`\begin{table}[`*place*`]`	Add numbered table.
`\begin{figure}[`*place*`]`	Add numbered figure.
`\begin{equation}[`*place*`]`	Add numbered equation.
`\caption{`*text*`}`	Caption for the body.

The *place* is a list valid placements for the body. `t`=top, `h`=here, `b`=bottom, `p`=separate page, `!`=place even if ugly. Captions and label markers should be within the environment.

Text properties

Font face

Command	*Declaration*	*Effect*
`\textrm{`*text*`}`	`{\rmfamily` *text*`}`	Roman family
`\textsf{`*text*`}`	`{\sffamily` *text*`}`	Sans serif family
`\texttt{`*text*`}`	`{\ttfamily` *text*`}`	`Typewriter family`
`\textmd{`*text*`}`	`{\mdseries` *text*`}`	Medium series
`\textbf{`*text*`}`	`{\bfseries` *text*`}`	**Bold series**
`\textup{`*text*`}`	`{\upshape` *text*`}`	Upright shape
`\textit{`*text*`}`	`{\itshape` *text*`}`	*Italic shape*
`\textsl{`*text*`}`	`{\slshape` *text*`}`	*Slanted shape*
`\textsc{`*text*`}`	`{\scshape` *text*`}`	Small Caps shape
`\emph{`*text*`}`	`{\em` *text*`}`	*Emphasized*
`\textnormal{`*text*`}`	`{\normalfont` *text*`}`	Document font
`\underline{`*text*`}`		Underline

The command (`ttt`) form handles spacing better than the declaration (*ttt*) form.

Font size

`\tiny`	tiny	`\Large` Large
`\scriptsize`	scriptsize	`\LARGE` LARGE
`\footnotesize`	footnotesize	`\huge` huge
`\small`	small	`\Huge` Huge
`\normalsize`	normalsize	
`\large`	large	

These are declarations and should be used in the form `{\small ...}`, or without braces to affect the entire document.

Verbatim text

`\begin{verbatim}`	Verbatim environment.
`\begin{verbatim*}`	Spaces are shown as ␣.
`\verb!text!`	Text between the delimiting characters (in this case '`!`') is verbatim.

Justification

Environment	*Declaration*
`\begin{center}`	`\centering`
`\begin{flushleft}`	`\raggedright`
`\begin{flushright}`	`\raggedleft`

Miscellaneous

`\linespread{`*x*`}` changes the line spacing by the multiplier *x*.

Text-mode symbols

Symbols

&	`\&`	_	`\_`	...	`\ldots`	•	`\textbullet`
$	`\$`	^	`\^{}`	|	`\textbar`	\	`\textbackslash`
%	`\%`	~	`\~{}`	#	`\#`	§	`\S`

Accents

ò	`\‘o`	ó	`\’o`	ô	`\^o`	ō	`\^o`	ō	`\=o`
ȯ	`\.o`	ö	`\"o`	o̧	`\c o`	ŏ	`\v o`	ő	`\H o`
ç	`\c c`	o̩	`\d o`	o̲	`\b o`	o͡o	`\t oo`	œ	`\oe`
Œ	`\OE`	æ	`\ae`	Æ	`\AE`	å	`\aa`	Å	`\AA`
ø	`\o`	Ø	`\O`	ł	`\l`	Ł	`\L`	ı	`\i`
ȷ	`\j`	¡	`~‘`	¿	`?‘`				

Delimiters

‘	‘	“	‘‘	{	`\{`	[	`[`	(	`(`	<	`\textless`
’	’	”	’’	}	`\}`	]	`]`	)	`)`	>	`\textgreater`

Dashes

Name	*Source*	*Example*	*Usage*
hyphen	`-`	X-ray	In words.
en-dash	`--`	1–5	Between numbers.
em-dash	`---`	Yes—or no?	Punctuation.

Line and page breaks

`\\`	Begin new line without new paragraph.
`\\*`	Prohibit pagebreak after linebreak.
`\kill`	Don't print current line.
`\pagebreak`	Start new page.
`\noindent`	Do not indent current line.

Miscellaneous

`\today`	February 25, 2014.
`$\sim$`	Prints ∼ instead of `\~{}`, which makes ˜.
`~`	Space, disallow linebreak (`W.J.~Clinton`).
`\@.`	Indicate that the `.` ends a sentence when following an uppercase letter.
`\hspace{`*l*`}`	Horizontal space of length *l* (Ex: $l =$ `20pt`).
`\vspace{`*l*`}`	Vertical space of length *l*.
`\rule{`*w*`}{`*h*`}`	Line of width *w* and height *h*.

Tabular environments

`tabbing` environment

`\=` Set tab stop. `\>` Go to tab stop.

Tab stops can be set on "invisible" lines with `\kill` at the end of the line. Normally `\\` is used to separate lines.

(a) Page 1

Figure 2.1: LaTeX cheat sheet *(continued in the next page)*

tabular environment

```
\begin{array}[pos]{cols}
\begin{tabular}[pos]{cols}
\begin{tabular*}{width}[pos]{cols}
```

tabular column specification

`l`	Left-justified column.
`c`	Centered column.
`r`	Right-justified column.
`p{`*width*`}`	Same as `\parbox[t]{`*width*`}`.
`@{`*decl*`}`	Insert *decl* instead of inter-column space.
`|`	Inserts a vertical line between columns.

tabular elements

`\hline`	Horizontal line between rows.
`\cline{`x-y`}`	Horizontal line across columns x through y.
`\multicolumn{`n`}{`*cols*`}{`*text*`}`	A cell that spans n columns, with *cols* column specification.

Math mode

For inline math, use `\(...\)` or `$...$`. For displayed math, use `\[...\]` or `\begin{equation}`.

Superscript x^y	`^{x}`	Subscript x_y	`_{x}`
$\frac{x}{y}$	`\frac{x}{y}`	$\sum_{k=1}^{n}$	`\sum_{k=1}^n`
$\sqrt[n]{x}$	`\sqrt[n]{x}`	$\prod_{k=1}^{n}$	`\prod_{k=1}^n`

Math-mode symbols

$\leq$	`\leq`	$\geq$	`\geq`	$\neq$	`\neq`	$\approx$	`\approx`
$\times$	`\times`	$\div$	`\div`	$\pm$	`\pm`	$\cdot$	`\cdot`
$\circ$	`^{\circ}`	$\circ$	`\circ`	$\prime$	`\prime`	$\cdots$	`\cdots`
∞	`\infty`	$\neg$	`\neg`	$\wedge$	`\wedge`	$\vee$	`\vee`
$\supset$	`\supset`	$\forall$	`\forall`	$\in$	`\in`	$\rightarrow$	`\rightarrow`
$\subset$	`\subset`	$\exists$	`\exists`	$\notin$	`\notin`	$\Rightarrow$	`\Rightarrow`
$\cup$	`\cup`	$\cap$	`\cap`	$\mid$	`\mid`	$\Leftrightarrow$	`\Leftrightarrow`
$\dot{a}$	`\dot a`	$\hat{a}$	`\hat a`	$\bar{a}$	`\bar a`	$\tilde{a}$	`\tilde a`
α	`\alpha`	β	`\beta`	γ	`\gamma`	δ	`\delta`
ϵ	`\epsilon`	ζ	`\zeta`	η	`\eta`	ε	`\varepsilon`
θ	`\theta`	ι	`\iota`	κ	`\kappa`	ϑ	`\vartheta`
λ	`\lambda`	μ	`\mu`	ν	`\nu`	ξ	`\xi`
π	`\pi`	ρ	`\rho`	σ	`\sigma`	τ	`\tau`
υ	`\upsilon`	ϕ	`\phi`	χ	`\chi`	ψ	`\psi`
ω	`\omega`	Γ	`\Gamma`	Δ	`\Delta`	Θ	`\Theta`
Λ	`\Lambda`	Ξ	`\Xi`	Π	`\Pi`	Σ	`\Sigma`
Υ	`\Upsilon`	Φ	`\Phi`	Ψ	`\Psi`	Ω	`\Omega`

Bibliography and citations

When using BibTeX, you need to run `latex`, `bibtex`, and `latex` twice more to resolve dependencies.

Citation types

`\cite{`*key*`}`	Full author list and year. (Watson and Crick 1953)
`\citeA{`*key*`}`	Full author list. (Watson and Crick)
`\citeN{`*key*`}`	Full author list and year. Watson and Crick (1953)
`\shortcite{`*key*`}`	Abbreviated author list and year. ?
`\shortciteA{`*key*`}`	Abbreviated author list. ?
`\shortciteN{`*key*`}`	Abbreviated author list and year. ?
`\citeyear{`*key*`}`	Cite year only. (1953)

All the above have an `NP` variant without parentheses; Ex. `\citeNP`.

BibTeX entry types

`@article`	Journal or magazine article.
`@book`	Book with publisher.
`@booklet`	Book without publisher.
`@conference`	Article in conference proceedings.
`@inbook`	A part of a book and/or range of pages.
`@incollection`	A part of book with its own title.
`@misc`	If nothing else fits.
`@phdthesis`	PhD. thesis.
`@proceedings`	Proceedings of a conference.
`@techreport`	Tech report, usually numbered in series.
`@unpublished`	Unpublished.

BibTeX fields

`address`	Address of publisher. Not necessary for major publishers.
`author`	Names of authors, of format ....
`booktitle`	Title of book when part of it is cited.
`chapter`	Chapter or section number.
`edition`	Edition of a book.
`editor`	Names of editors.
`institution`	Sponsoring institution of tech. report.
`journal`	Journal name.
`key`	Used for cross ref. when no author.
`month`	Month published. Use 3-letter abbreviation.
`note`	Any additional information.
`number`	Number of journal or magazine.
`organization`	Organization that sponsors a conference.
`pages`	Page range (`2,6,9--12`).
`publisher`	Publisher's name.
`school`	Name of school (for thesis).
`series`	Name of series of books.
`title`	Title of work.
`type`	Type of tech. report, ex. "Research Note".
`volume`	Volume of a journal or book.
`year`	Year of publication.

Not all fields need to be filled. See example below.

Common BibTeX style files

`abbrv`	Standard	`abstract`	`alpha` with abstract
`alpha`	Standard	`apa`	APA
`plain`	Standard	`unsrt`	Unsorted

The LaTeX document should have the following two lines just before `\end{document}`, where `bibfile.bib` is the name of the BibTeX file.

```
\bibliographystyle{plain}
\bibliography{bibfile}
```

BibTeX example

The BibTeX database goes in a file called *file*`.bib`, which is processed with `bibtex file`.

```
@String{N = {Na\-ture}}
@Article{WC:1953,
  author  = {James Watson and Francis Crick},
  title   = {A structure for Deoxyribose Nucleic Acid},
  journal = N,
  volume  = {171},
  pages   = {737},
  year    = 1953
}
```

Sample LaTeX document

```
\documentclass[11pt]{article}
\usepackage{fullpage}
\title{Template}
\author{Name}
\begin{document}
\maketitle

\section{section}
\subsection*{subsection without number}
text \textbf{bold text} text. Some math: $2+2=5$
\subsection{subsection}
text \emph{emphasized text} text. \cite{WC:1953}
discovered the structure of DNA.

A table:
\begin{table}[!th]
\begin{tabular}{|l|c|r|}
\hline
first  &  row  &  data \\
second &  row  &  data \\
\hline
\end{tabular}
\caption{This is the caption}
\label{ex:table}
\end{table}

The table is numbered \ref{ex:table}.
\end{document}
```

Copyright © 2014 Winston Chang
http://www.stdout.org/~winston/latex/

(b) Page 2

Figure 2.1: *(continued)* LaTeX cheat sheet

referenced files as well.

2.3 Arabic Support

Thanks to[1] the "Arabi" package, Arabic and Farsi languages are supported with the "Babel" package.

However, since Arabic users are few, "Arabi" package is not mature enough and some minor bugs do exist. Googling about these bugs, usually you find the similar bugs do exist in other languages as well, and hence you can infer solutions/workarounds. During preparing this thesis, I have done my best to solve/work-around all the bugs I have faced.

2.4 Installing LaTeX

1. Install LaTeX implementation. Notable implementations are:
   - **MiKTeX** Windows only[2]
   - **TeX Live** cross-platform[3]
2. Install TeX/LaTeX editor/IDE. Notable examples include Texstudio, Kile, ...

Keep Concentrating

Due to its WYSIWYM nature, I feel more concentrating while using **LaTeX** as compared to **Ms-Word**

[1] Thanks to GOD at first of course.

[2] Download the full MiKTeX. This is done using the "**Net Installer**". First, download the full MiKTeX. After download completes, run the downloaded installer and install the full MiKTeX.

[3] Available for MS-Windows, Mac OS and Linux

Chapter 3

LyX; a Graphical Front-End to LaTeX

LyX is a graphical front-end to LaTeX

- You can think of the LyX-LaTeX relationship as similar to the Visual Studio-C++ compiler relationship
- Unlike LaTeX, LyX comes with tidy and very good documentation
- Also it has a big community, i.e.,
  - it is mature enough
  - when you encounter a problem, google it. Most likely you will find others had encountered it and found a solution

Keep your concentration

Due to its WYSIWYM nature, I feel very concentrating while using **LyX** as compared to **Ms-Word**.

3.1 Installing LyX

1. Install Inkscape
   - Confirm path to inkscape.exe is added to the "PATH" environment variable
2. Install the ***full*** MiKTeX (or TeX Live)
3. Install LyX
   - **Windows**
     - Installers are available at `www.lyx.org`
   - **Linux**
     - LyX is usually available in most Linux distributions' repositories
     - To receive the latest stable updates in **Ubuntu**, add the following ppa by executing the following shell command

       ```
       > sudo add-apt-repository ppa:lyx-devel/release
       ```

4. Modify LyX configurations to use Inkscape as graphics translator, as explained in figure 3.1. That is, Tools ▷ Preferences ▷ Converters[1] ▷
   - SVG -> EPS > Converter > `inkscape $$i --export-area-drawing --export-type="eps"`

[1] Note that Inkscape CLI has changed since version 1.0 [https://wiki.inkscape.org/wiki/index.php/Using_the_Command_Line#Changes_from_0.92]

- `SVG -> PDF > Converter > inkscape $$i --export-area-drawing --export-type="pdf"`
- `SVG -> PNG > Converter > inkscape $$i --export-type="png"`
- `GIF -> PNG > Converter > magick convert '$$i[0]' $$o`[1]

5. Enable continuous spell checking
   Tools ▷ Preferences ▷ Language Settings ▷ Spellchecker ▷ Spellcheck continuously

3.2 Learning LyX

Explore style-list, menus and toolbars
Help menu includes very good manuals

- Manuals themselves are LyX documents
  - So they are essentially very good LyX examples
- You may begin with:
  1. Introduction
  2. Tutorial
- Then if needed, read necessary sections of:
  1. User's Guide
  2. rest of manuals ...

lyx\examples folder contains wide variety of very good examples

3.3 Porting a LyX Document

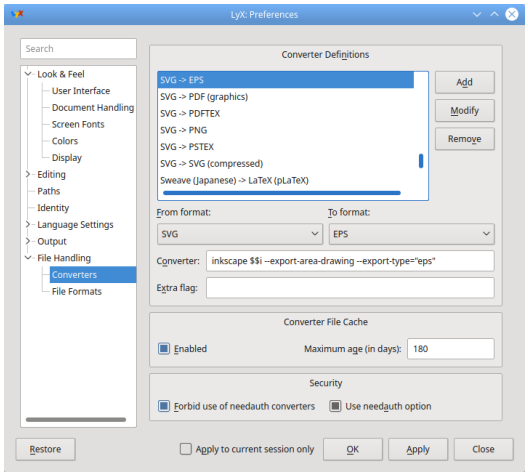
Similar to LaTeX files, LyX files usually reference images and other external files. Hence, if you want to move/copy your LyX document to another computer, you have to move/copy all the referenced files as well.

LyX greatly simplifies collecting the referenced files by the command LyX ▷ File ▷ Export ▷ LyX Archive

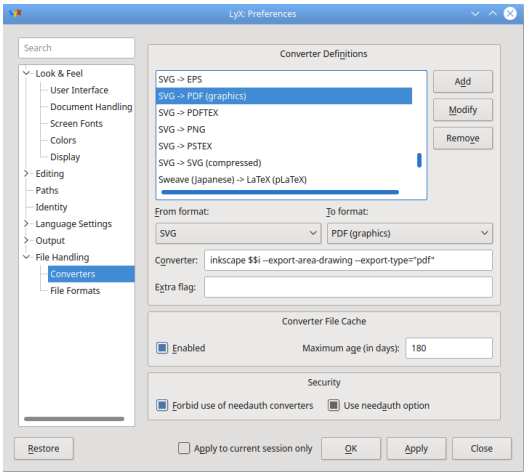
3.4 Arabic Support

Arabic is supported in LyX, as shown in the following. For more details, refer to section 2.3.

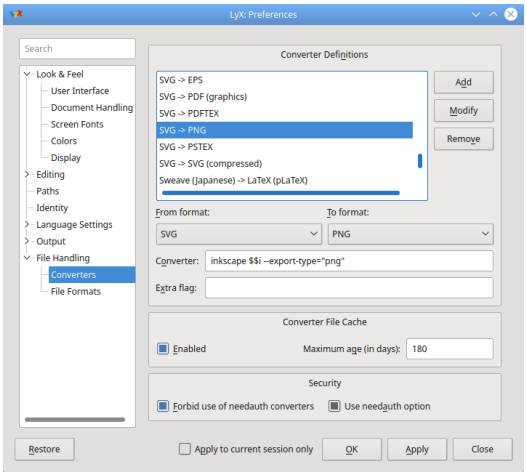
[1] For ImageMagick older than release 7.x, use "`convert '$$i[0]' $$o`"



(a) To convert svg to eps



(b) To convert svg to pdf



(c) To convert svg to png

Figure 3.1: Correcting svg converters in Inkscape

This page is intentionally left blank!

Chapter 4

Floats, Figures, Tables and Equations

4.1 Concept of Floating Graphics, Tables

For those users familiar with MS World, they expect figures and tables are placed where you put them. This however does not look professional. Therefore, LaTeX, and consequently LyX, uses floats for placing figures and tables. Sample simple floating figures are figures 1.1 and 7.1.

For more information about this topic, refer to [2] and [3, sec. 4.6].

4.2 Compound Figures

Figures composed of sub-figures can be created in by using the subcaption LaTeX package. Sample compound figures are figures 2.1, 3.1, 4.1, 6.1, 6.2, 7.2 and 7.3.

4.2.1 Subfigure and Subtable

Have a look at figure 4.1.

4.3 Continued Floats

Figure 2.1 shows a sample float continued from a float to another.

4.4 Landscape Floats

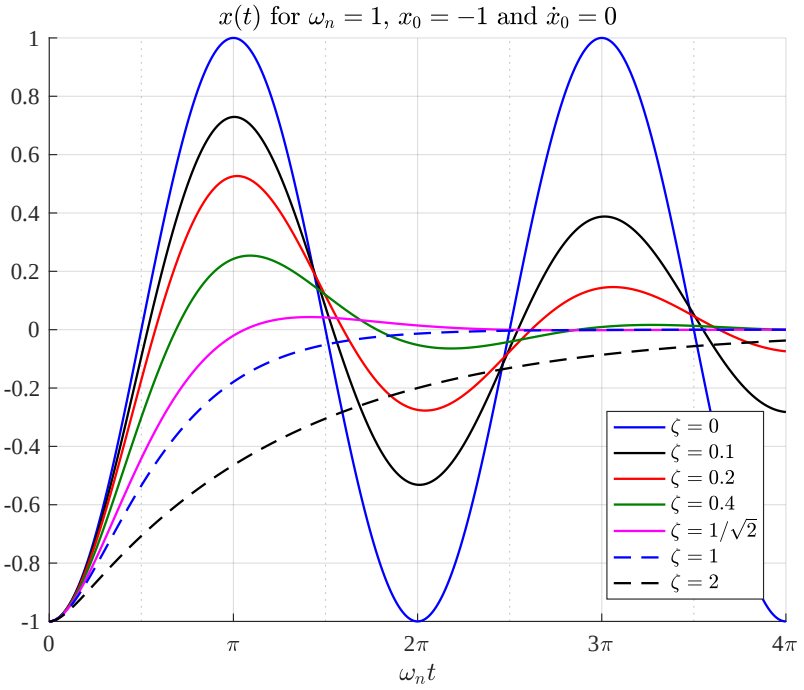
Have a look at figure 2.1.

4.5 Side-by-Side Facing Floats

Have a look at figures 6.1 and 6.2.

4.6 Free Inline Graphics without Captions

Have a look at graphics of chapter 10.



(a) Free vibration of a SDOF system

ρ_{ij}	$i = 1$	$i = 2$	$i = 3$	$i = 4$
$j = 1$	1.0000	-0.0000	-0.8328	-0.0010
$j = 2$	-0.0000	1.0000	-0.0000	-0.8328
$j = 3$	-0.8328	-0.0000	1.0000	-0.0000
$j = 4$	-0.0010	-0.8328	-0.0000	1.0000

(b) Table with numbers aligned at the decimal point and formatted as "typewriter"

Figure 4.1: Figure composed of a subfigure and subtable

4.7 Tables

Table 4.1 shows a sample simple table, while table 4.2 shows a more complex table. Additional details are available in [3, sec. 4.5] and [2, chapter 2].

4.8 Equations

For details about equations, refer to [4]. The following is sample text with various types of equations.

4.8.1 SDOF Mass Spring System

Table 4.1: Table caption

	Conventional Transducer	This Transducer
Price	word word	word word
Size	word word	word word
Weight	word word	word word
Coupling	word word	word word
Material	word word	word word
Generation	word word	word word
Suitability	word word	word word
Restrictions	word word	word word
Action type	word word	word word

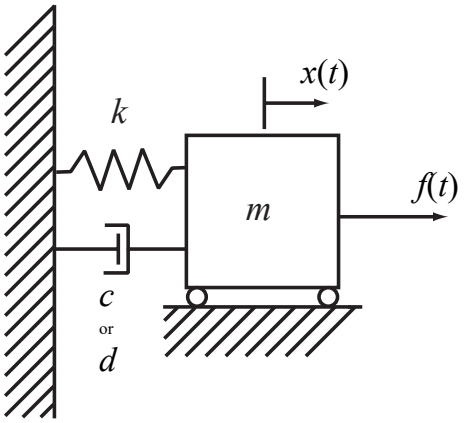


Figure 4.2: SDOF Mass Spring System

Governing Ordinary Differential Equation (ode)

$$m\ddot{x}(t) + c\dot{x}(t) + kx(t) = f(t) \tag{4.1}$$

Taking Laplace transform, the *ode* is transformed to the algebraic equation

$$m\left(s^2X(s) - sx_0 - \dot{x}_0\right) + c\left(sX(s) - x_0\right) + kX(s) = F(s)$$

where $x_0 \equiv x(t=0)$ and $\dot{x}_0 \equiv \dot{x}(t=0)$.

Rearranging yields

$$\left(ms^2 + cs + k\right)X(s) - \left(ms + c\right)x_0 - m\dot{x}_0 = F(s) \tag{4.2}$$

Dividing by m yields

$$\left(s^2 + 2\zeta\omega_{\text{n}}s + \omega_{\text{n}}^2\right)X(s) - \left(s + 2\zeta\omega_{\text{n}}\right)x_0 - \dot{x}_0 = \frac{F(s)}{m} \tag{4.3}$$

where the non-dimensional parameters ω_{n} and ζ are the ***natural frequency*** and ***damping ratio*** defined as

$$\boxed{\omega_{\text{n}} \equiv \sqrt{\frac{k}{m}}} \qquad \& \qquad \boxed{\zeta \equiv \frac{c}{c_{\text{c}}}} \tag{4.4}$$

Table 4.2: Comparison between somethings

	Type 1	Type 2	Type 3	Type 4
Feature 1	words words words words words words words	words words words words words words words	words words words words words words words	words words words words words words words
Feature 2	words words words words words words words	words words words words words words words	words words words words words words words	words words words words words words words
Feature 3	words words words words words words words	words words words words words words words	words words words words words words words	words words words words words words words
Feature 4	words words words words words words words	words words words words words words words	words words words words words words words	words words words words words words words

where c_c is the ***critical damping*** defined as

$$\boxed{c_\mathrm{c} \equiv 2\sqrt{km}} \tag{4.5}$$

By solving the algebraic equation (4.3), the response $X(s)$ is obtained as

$$\begin{aligned} X(s) &= \frac{F(s)}{m\left(s^2+2\zeta\omega_\mathrm{n}s+\omega_\mathrm{n}^2\right)} \\ &\quad + \frac{sx_0}{s^2+2\zeta\omega_\mathrm{n}s+\omega_\mathrm{n}^2} + \frac{2\zeta\omega_\mathrm{n}x_0+\dot{x}_0}{s^2+2\zeta\omega_\mathrm{n}s+\omega_\mathrm{n}^2} \end{aligned}$$

or

$$X(s) = F(s)\,H(s) + \frac{sx_0}{s^2+2\zeta\omega_\mathrm{n}s+\omega_\mathrm{n}^2} + \frac{2\zeta\omega_\mathrm{n}x_0+\dot{x}_0}{s^2+2\zeta\omega_\mathrm{n}s+\omega_\mathrm{n}^2} \tag{4.6}$$

where $H(s)$ is the *Transfer Function* (TF) defined as

$$H(s) \equiv \frac{X(s)|_{\text{zero initial conditions}}}{F(s)} \tag{4.7}$$

$$= \frac{1}{ms^2+cs+k} \tag{4.8}$$

$$= \frac{1}{m\left(s^2+2\zeta\omega_\mathrm{n}s+\omega_\mathrm{n}^2\right)} \tag{4.9}$$

$$= \frac{1}{m\left(s-\left(-\zeta\omega_\mathrm{n}+\omega_\mathrm{n}\sqrt{\zeta^2-1}\right)\right)\left(s-\left(-\zeta\omega_\mathrm{n}-\omega_\mathrm{n}\sqrt{\zeta^2-1}\right)\right)} \tag{4.10}$$

Assuming the roots of $H(s)$ are complex, the TF is written as

$$H(s) = \frac{1}{m\left(s-\left(-\zeta\omega_\mathrm{n}+i\omega_\mathrm{n}\sqrt{1-\zeta^2}\right)\right)\left(s-\left(-\zeta\omega_\mathrm{n}-i\omega_\mathrm{n}\sqrt{1-\zeta^2}\right)\right)} \tag{4.11}$$

or

$$\boxed{H(s) = \frac{1}{m\left(s - \left(-\zeta\omega_{\mathrm{n}} + i\omega_{\mathrm{d}}\right)\right)\left(s - \left(-\zeta\omega_{\mathrm{n}} - i\omega_{\mathrm{d}}\right)\right)}} \tag{4.12}$$

where

$$\boxed{\omega_{\mathrm{d}} \equiv \omega_{\mathrm{n}}\sqrt{1-\zeta^2}} \tag{4.13}$$

Thus the response $x(t)$ can be obtained from equation (4.6) as

$$x(t) = \mathcal{L}^{-1}\left[X(s)\right] \tag{4.14}$$

where $\mathcal{L}^{-1}$ denotes inverse Laplace transform.

Assuming the TF roots are complex, i.e., $\zeta < 1$, inverse Laplace transform tables yield

$$\begin{aligned} x(t) =& \mathcal{L}^{-1}\left[F(s)\,H(s)\right] \\ &+ x_0 e^{-\zeta\omega_{\mathrm{n}}t}\left(\cos\left(\omega_{\mathrm{d}}t\right) - \frac{\zeta\omega_{\mathrm{n}}}{\omega_{\mathrm{d}}}\sin\left(\omega_{\mathrm{d}}t\right)\right) \\ &+ \left(2\zeta\omega_{\mathrm{n}}x_0 + \dot{x}_0\right)\, e^{-\zeta\omega_{\mathrm{n}}t}\frac{\sin\left(\omega_{\mathrm{d}}t\right)}{\omega_{\mathrm{d}}} \end{aligned} \tag{4.15}$$

Rearranging yields

$$\boxed{\begin{aligned} x(t) =& \mathcal{L}^{-1}\left[F(s)\,H(s)\right] \\ &+ e^{-\zeta\omega_{\mathrm{n}}t}\left[x_0\cos\left(\omega_{\mathrm{d}}t\right) + \left(\zeta\omega_{\mathrm{n}}x_0 + \dot{x}_0\right)\frac{\sin\left(\omega_{\mathrm{d}}t\right)}{\omega_{\mathrm{d}}}\right] \end{aligned}} \tag{4.16}$$

or from the convolution property

$$\begin{aligned} x(t) =& \left(f * h\right)(t) \\ &+ e^{-\zeta\omega_{\mathrm{n}}t}\left[x_0\cos\left(\omega_{\mathrm{d}}t\right) + \left(\zeta\omega_{\mathrm{n}}x_0 + \dot{x}_0\right)\frac{\sin\left(\omega_{\mathrm{d}}t\right)}{\omega_{\mathrm{d}}}\right] \end{aligned} \tag{4.17}$$

where

$$\boxed{h(t) \equiv \mathcal{L}^{-1}\left[H(s)\right] = \frac{e^{-\zeta\omega_{\mathrm{n}}t}}{m}\frac{\sin\left(\omega_{\mathrm{d}}t\right)}{\omega_{\mathrm{d}}}} \tag{4.18}$$

is the Impulse Response Function (IRF), and

$$\left(f * h\right)(t) \equiv \int_{-\infty}^{\infty} f(\tau)h(t-\tau)\,d\tau \tag{4.19}$$

$$= \int_{0}^{t} f(\tau)h(t-\tau)\,d\tau \quad ,: f(t) = h(t) = 0\ \forall t < 0 \tag{4.20}$$

is the convolution of $f(t)$ and $h(t)$, assuming stable, linear, physically possible and time invariant system.

4.8.2 Inverse Laplace Transform Derivation

Using Laplace transform property, inverse Laplace can be obtained as

$$\frac{\Omega\, s}{\left(s^2 + \Omega^2\right)\left(s^2 + 2\zeta\omega_{\mathrm{n}}s + \omega_{\mathrm{n}}^2\right)} \overset{\mathcal{L}}{\Longleftrightarrow} \dot{y}(t) + y(0) \tag{4.21}$$

where $y(t)$ is the inverse Laplace transform of

$$\frac{\Omega}{\left(s^2+\Omega^2\right)\left(s^2+2\zeta\omega_{\mathrm{n}}s+\omega_{\mathrm{n}}^2\right)}$$

previously derived as

$$y(t)=\frac{-2\zeta r\,\cos\left(\Omega t\right)+\left(1-r^2\right)\sin\left(\Omega t\right)+r\,e^{-\zeta\omega_{\mathrm{n}}t}\left[2\zeta\,\cos\left(\omega_{\mathrm{d}}t\right)+\omega_{\mathrm{n}}\left(2\zeta^2-\left(1-r^2\right)\right)\frac{\sin(\omega_{\mathrm{d}}t)}{\omega_{\mathrm{d}}}\right]}{\omega_{\mathrm{n}}^2\left(\left(1-r^2\right)^2+\left(2\zeta r\right)^2\right)} \tag{4.22}$$

Thus

$$y(0)=\frac{-2\zeta r+2\zeta r}{\omega_{\mathrm{n}}^2\left(\left(1-r^2\right)^2+\left(2\zeta r\right)^2\right)}=0 \tag{4.23}$$

and

$$\begin{aligned}
\dot{y}(t)=&\frac{\Omega}{\omega_{\mathrm{n}}^2}\,\frac{2\zeta r\,\sin\left(\Omega t\right)+\left(1-r^2\right)\cos\left(\Omega t\right)}{\left(1-r^2\right)^2+\left(2\zeta r\right)^2}+\frac{r}{\omega_{\mathrm{n}}^2\left(\left(1-r^2\right)^2+\left(2\zeta r\right)^2\right)}\\
&\times\left[\omega_{\mathrm{d}}e^{-\zeta\omega_{\mathrm{n}}t}\left(-2\zeta\,\sin\left(\omega_{\mathrm{d}}t\right)+\omega_{\mathrm{n}}\left(2\zeta^2-\left(1-r^2\right)\right)\frac{\cos\left(\omega_{\mathrm{d}}t\right)}{\omega_{\mathrm{d}}}\right)\right.\\
&\qquad\left.-\zeta\omega_{\mathrm{n}}e^{-\zeta\omega_{\mathrm{n}}t}\left(2\zeta\,\cos\left(\omega_{\mathrm{d}}t\right)+\omega_{\mathrm{n}}\left(2\zeta^2-\left(1-r^2\right)\right)\frac{\sin\left(\omega_{\mathrm{d}}t\right)}{\omega_{\mathrm{d}}}\right)\right]\\
=&\frac{r}{\omega_{\mathrm{n}}}\,\frac{\left(1-r^2\right)\cos\left(\Omega t\right)+2\zeta r\,\sin\left(\Omega t\right)}{\left(1-r^2\right)^2+\left(2\zeta r\right)^2}+\frac{r}{\omega_{\mathrm{n}}^2\left(\left(1-r^2\right)^2+\left(2\zeta r\right)^2\right)}\\
&\times\left[\left(\omega_{\mathrm{n}}\left(2\zeta^2-\left(1-r^2\right)\right)-2\zeta^2\omega_{\mathrm{n}}\right)\cos\left(\omega_{\mathrm{d}}t\right)\right.\\
&\qquad\left.+\left(-2\zeta\omega_{\mathrm{d}}-\frac{\zeta\omega_{\mathrm{n}}^2\left(2\zeta^2-(1-r^2)\right)}{\omega_{\mathrm{d}}}\right)\sin\left(\omega_{\mathrm{d}}t\right)\right]\\
=&\frac{r}{\omega_{\mathrm{n}}}\,\frac{\left(1-r^2\right)\cos\left(\Omega t\right)+2\zeta r\,\sin\left(\Omega t\right)}{\left(1-r^2\right)^2+\left(2\zeta r\right)^2}+\frac{re^{-\zeta\omega_{\mathrm{n}}t}}{\omega_{\mathrm{n}}^2\left(\left(1-r^2\right)^2+\left(2\zeta r\right)^2\right)}\\
&\times\left[-\omega_{\mathrm{n}}\left(1-r^2\right)\,\cos\left(\omega_{\mathrm{d}}t\right)\right.\\
&\qquad\left.+\left(-2\zeta\omega_{\mathrm{d}}^2-\zeta\omega_{\mathrm{n}}^2\left(2\zeta^2-\left(1-r^2\right)\right)\right)\frac{\sin\left(\omega_{\mathrm{d}}t\right)}{\omega_{\mathrm{d}}}\right]\\
=&\frac{r}{\omega_{\mathrm{n}}}\,\frac{\left(1-r^2\right)\cos\left(\Omega t\right)+2\zeta r\,\sin\left(\Omega t\right)}{\left(1-r^2\right)^2+\left(2\zeta r\right)^2}+\frac{re^{-\zeta\omega_{\mathrm{n}}t}}{\omega_{\mathrm{n}}^2\left(\left(1-r^2\right)^2+\left(2\zeta r\right)^2\right)}\\
&\times\left[-\omega_{\mathrm{n}}\left(1-r^2\right)\,\cos\left(\omega_{\mathrm{d}}t\right)\right.\\
&\qquad\left.+\zeta\omega_{\mathrm{n}}^2\left(-2\left(1-\zeta^2\right)-2\zeta^2+\left(1-r^2\right)\right)\frac{\sin\left(\omega_{\mathrm{d}}t\right)}{\omega_{\mathrm{d}}}\right]\\
=&\frac{r}{\omega_{\mathrm{n}}}\,\frac{\left(1-r^2\right)\cos\left(\Omega t\right)+2\zeta r\,\sin\left(\Omega t\right)}{\left(1-r^2\right)^2+\left(2\zeta r\right)^2}+\frac{re^{-\zeta\omega_{\mathrm{n}}t}}{\omega_{\mathrm{n}}^2\left(\left(1-r^2\right)^2+\left(2\zeta r\right)^2\right)}\\
&\times\left[-\omega_{\mathrm{n}}\left(1-r^2\right)\,\cos\left(\omega_{\mathrm{d}}t\right)+\zeta\omega_{\mathrm{n}}^2\left(-2+\left(1-r^2\right)\right)\frac{\sin\left(\omega_{\mathrm{d}}t\right)}{\omega_{\mathrm{d}}}\right]\\
=&\frac{r}{\omega_{\mathrm{n}}}\,\frac{\left(1-r^2\right)\cos\left(\Omega t\right)+2\zeta r\,\sin\left(\Omega t\right)}{\left(1-r^2\right)^2+\left(2\zeta r\right)^2}+\frac{re^{-\zeta\omega_{\mathrm{n}}t}}{\omega_{\mathrm{n}}\left(\left(1-r^2\right)^2+\left(2\zeta r\right)^2\right)}
\end{aligned}$$

$$\times \left[-\left(1-r^2\right)\,\cos\left(\omega_{\mathrm{d}}t\right) - \zeta\omega_{\mathrm{n}}\left(1+r^2\right)\frac{\sin\left(\omega_{\mathrm{d}}t\right)}{\omega_{\mathrm{d}}} \right] \tag{4.24}$$

Substituting equations (4.23) and (4.24) in (4.21) yields

$$\boxed{\begin{aligned} &\frac{\Omega\, s}{\left(s^2+\Omega^2\right)\left(s^2+2\zeta\omega_{\mathrm{n}}s+\omega_{\mathrm{n}}^2\right)} \overset{\mathcal{L}}{\Longleftrightarrow} \\ &\frac{r}{\omega_{\mathrm{n}}}\,\frac{\left(1-r^2\right)\cos\left(\Omega t\right)+2\zeta r\,\sin\left(\Omega t\right)-e^{-\zeta\omega_{\mathrm{n}}t}\left[\left(1-r^2\right)\,\cos\left(\omega_{\mathrm{d}}t\right)+\zeta\omega_{\mathrm{n}}\left(1+r^2\right)\frac{\sin(\omega_{\mathrm{d}}t)}{\omega_{\mathrm{d}}}\right]}{\left(1-r^2\right)^2+\left(2\zeta r\right)^2} \end{aligned}} \tag{4.25}$$

This page is intentionally left blank!

Chapter 5

Reference Management Software

Reference management software [5] is citation management software or personal bibliographic management software is software for scholars and authors to use for recording and utilising bibliographic citations (references) [6]. Once a citation has been recorded, it can be used time and again in generating bibliographies, such as lists of references in scholarly books, articles and essays. The development of reference management packages has been driven by the rapid expansion of scientific literature. Among poular refence management software are:

JabRef, a BibTeX management cross-platform software for use with LaTeX/LyX.

Endnote, a management software suitable for use with MS Word

Zotero, a cross-platform web-based management software suitable for LaTeX/LyX, MS Word, LibreOffice and others.

Comparisons of these software are available in [7].

This page is intentionally left blank!

Chapter 6

Vector Graphics

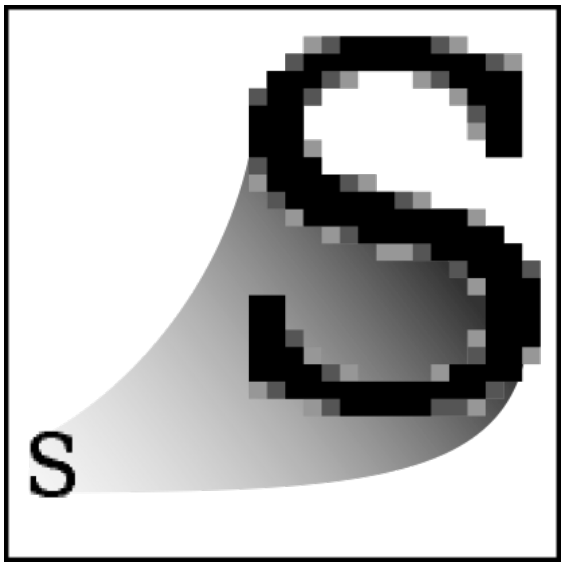
6.1 Raster vs Vector Graphics

Graphics Formats

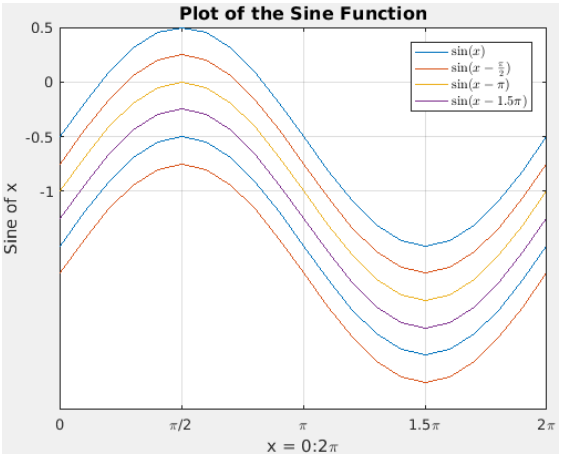
Raster file formats		Vector file formats	
.bmp	Uncompressed	.eps	
.png	Loose-less compression	.pdf	Compressed
.jpg	Lossy compression	.emf	Compatible with MS office
		.svg	Compatible with MS office & web browsers
⋮		⋮	

6.2 Vector Graphics Editors

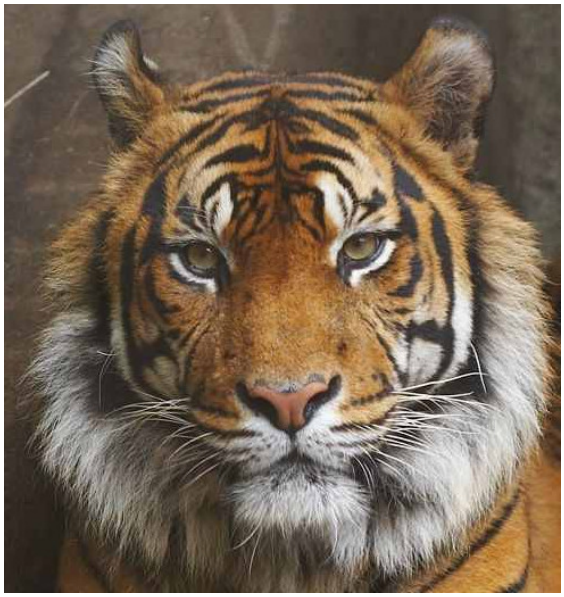
- Adobe Illustrator (*de facto* standard; bloated)
- Corel Draw (bloated)
- Inkscape (light, free, open source, cross-platform and popular; my favorite)
- LibreOffice Draw
- …



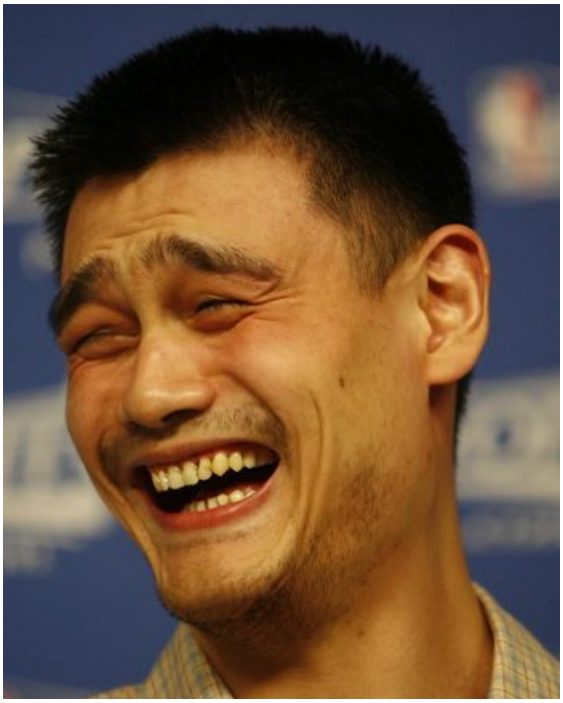
(a) Letter



(b) Matlab figure

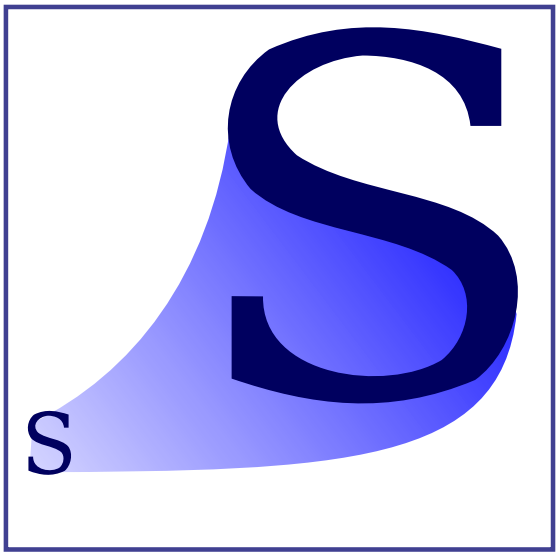


(c) Tiger

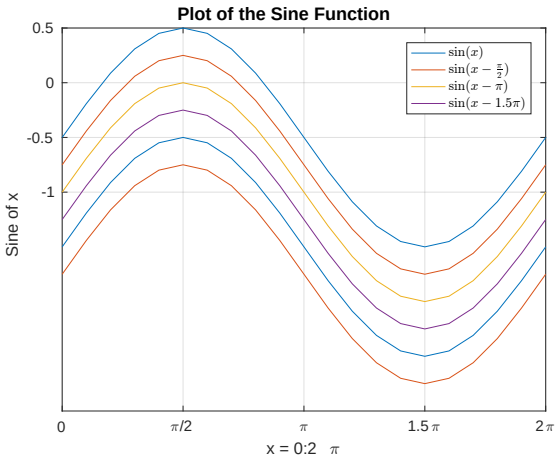


(d) Face

Figure 6.1: Sample raster graphics. This figure is forced to be on a left page for easier comparison with figure 6.2 on the opposite page.

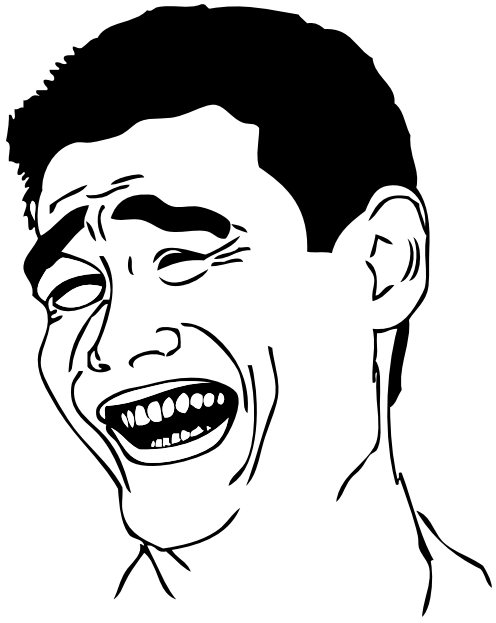


(a) Letter



(b) Matlab figure

(c) Tiger



(d) Face

Figure 6.2: Vector graphics version of figure 6.1

This page is intentionally left blank!

Chapter 7

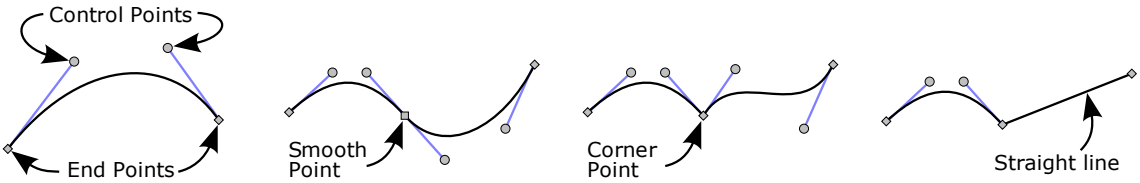
Inkscape; Free and Open Source Vector Graphics Editor

Inkscape Features

- FOSS
- Cross platform
- Much much powerful than MS-Word or MS-Power point sketching capabilities
- Has several plugins that greatly expand its capabilities
- Has a big community, i.e.,
  - it is mature enough
  - when you encounter a problem, google it. Most likely you will find others had encountered it and found a solution

Inkscape Capabilities

Inkscape is based on cubic Bézier[1] curves



- A curve is defined using two end points (nodes) and two control points (handles).
- Additionally, Inkscape can draw and edit:
  - straight lines
  - circles/arcs/ellipses
  - text
  - LaTeX formulas
  - function curves
  - …

7.1 Import Graphics from pdf

You can import vector graphics from pdf files, and even edit them, as shown in figure 7.1.

[1] [https://en.wikipedia.org/wiki/B%C3%A9zier_curve]

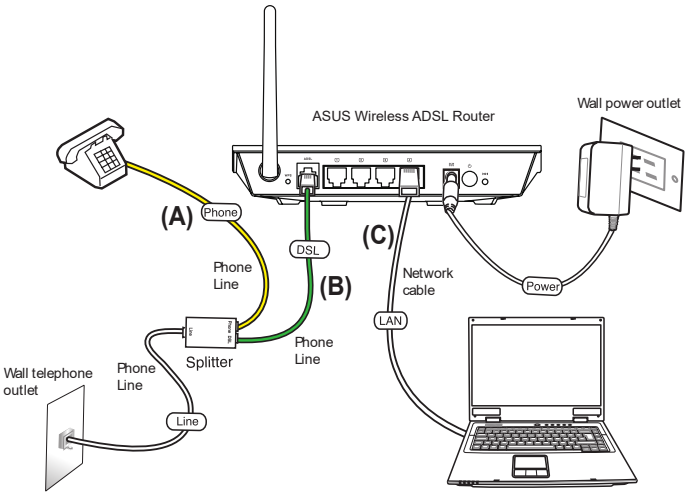
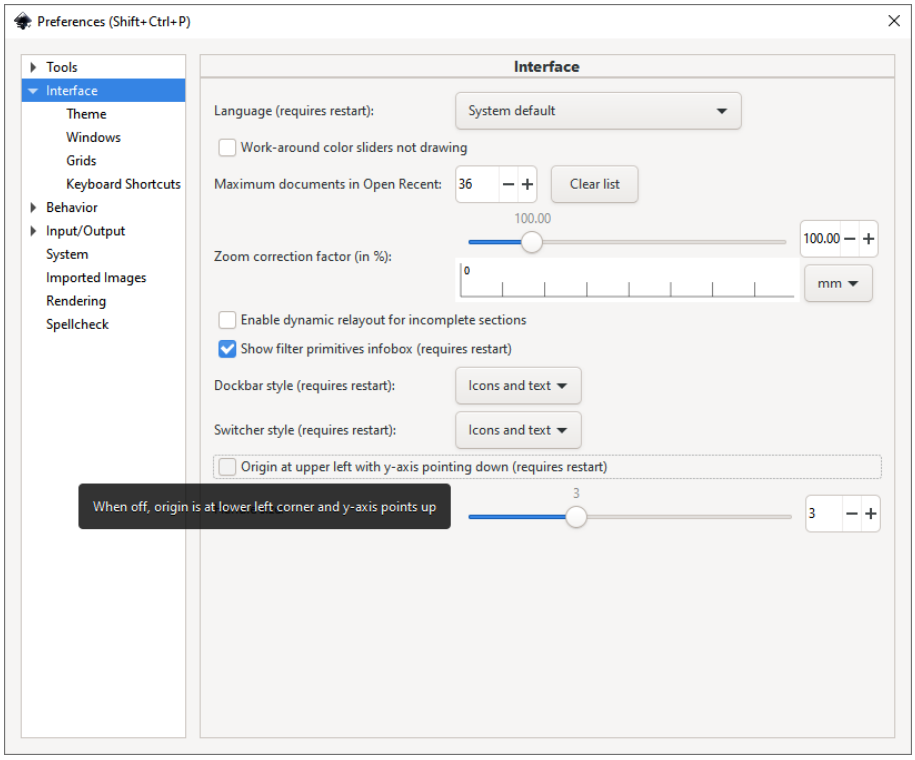
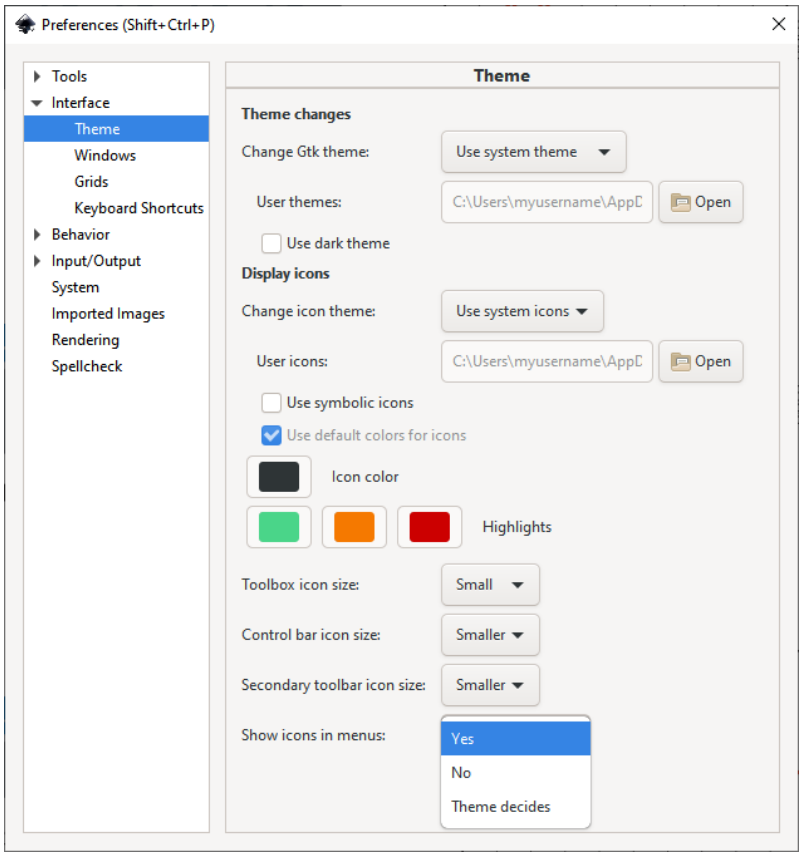


Figure 7.1: Vector graphic imported from the user guide of a home use ADSL router

7.2 Interesting Preferences





7.3 Interesting Plug-ins

7.3.1 Function Plotter

- It is a built in plugins
- It uses brazier curves, same as Inkscape
- It calculates the function derivative and use it to adjust the curve slope
  - It produces very smooth curves using much less points than Matlab
  - You can still adjust/correct the curve manually

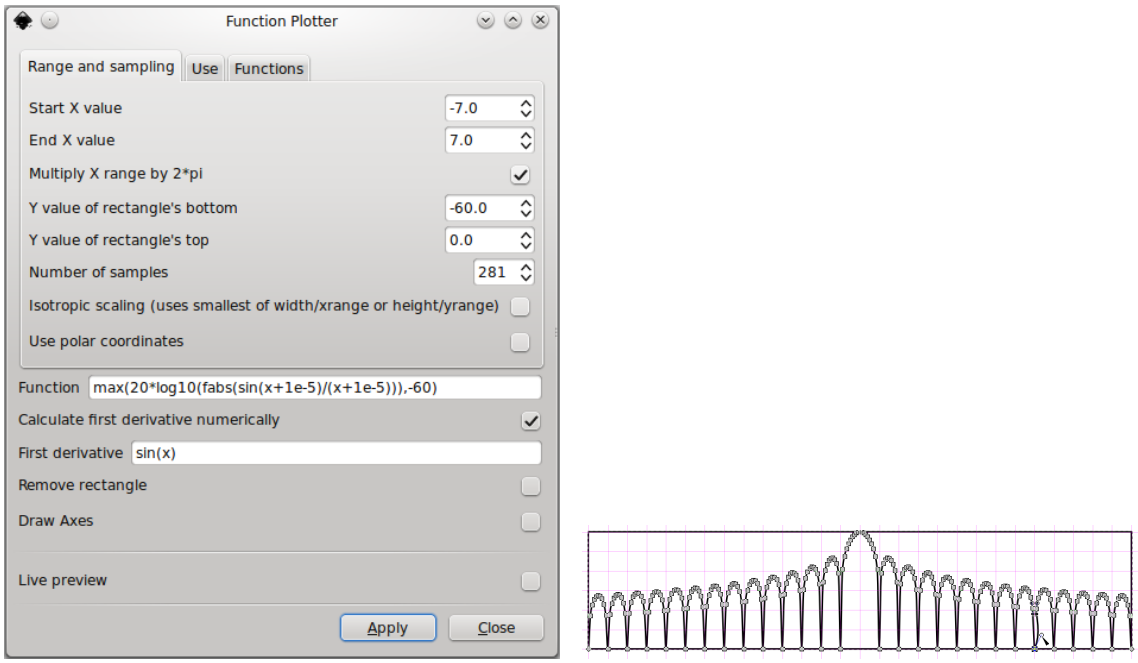
Figure 7.2 shows the plugin user interface, and the resulting curve. Figure 7.3 shows a more comprehensive example.

7.3.2 TexText

It allows you to write/edit LaTeX formulas inside Inkscape, as shown in figure 7.3.

7.4 Learning Inkscape

- **Explore** menus and toolbars
- **Official manual** [8] is very good and detailed
  - Chapters 1 includes 10 examples
    - The first 3 examples are enough for a good start



(a) Function Plotter user interface

(b) Curve generated by Function Plotter

Figure 7.2: The Function Plotter plugin

  - Chapters 5 explains editing
    - Surf it fast
- **Help menu** includes tutorials, FAQ, ...
- http://inkscapetutorials.org/

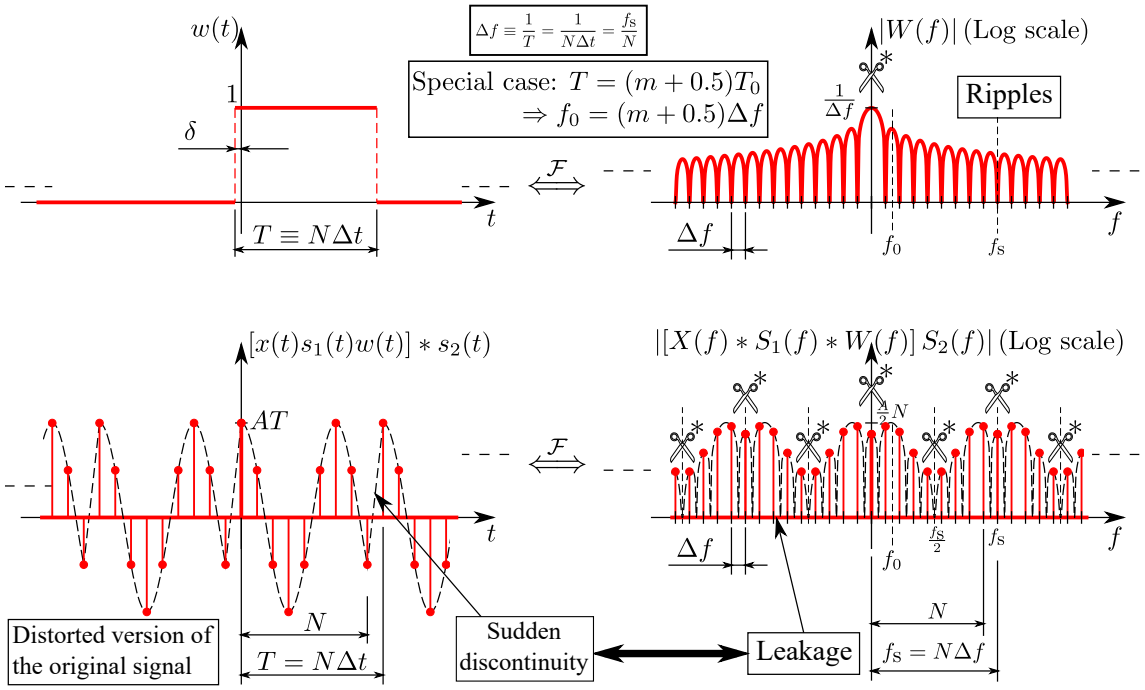


Figure 7.3: Figure illustrating the capabilities of “Function Plotter” and “TexText” plug ins.

This page is intentionally left blank!

Chapter 8

Including Program Codes

There is the listings LaTeX package which greatly simplifies adding program codes. Details are available in [2, chapter 8]. For example, codes A.1 and A.2 are used to plot figure 4.1(a).

Code A.3 on the other hand exports a Matlab figure a pdf file and crops it by removing white margins. Cropping is accomplished by calling a Perl program called "`pdfcrop`". This program, ships with both MiKTeX and TeX Live LaTeX implementations. To use this program, Perl is needed to be installed[1].

[1] "Strawberry Perl" is a sample open-source Perl implementation for Microsoft Windows.

This page is intentionally left blank!

Chapter 9

About the Nomenclature

If you defined a nomenclature entry twice, it results in an error (Lonely \item–perhaps a missing list environment.).

9.1 Problems with Arabic

Nomenclature (and may be index too) sometimes causes problems in Arabic documents. As a workaround (assuming your thesis file name is "Thesis"):

1. pdflatex the Thesis.tex file twice (or as needed)
2. manually edit the *.nlo file and modify as follows
   modify lines similar to this
   `\nomenclatureentry{aVI@[{VI}]\begingroup Visual Inspection\nomeqref {1.0}|nompageref}{\if@rlmain \I {1}\else \textLR {1}\fi }`
   to this
   `\nomenclatureentry{aVI@[{VI}]\begingroup Visual Inspection\nomeqref {1.0}|nompageref}{1}`
3. Run the command

   ```
   makeindex 'Thesis.nlo' -s nomencl.ist -o 'Thesis.nls'
   ```

4. pdflatex the Thesis.tex file once more (or as needed)

This page is intentionally left blank!

Chapter 10

Version Control Using Git

You can think of a Version[1] Control System (VCS) as a kind of "database" [1]. It lets you save a snapshot of your complete project at any time you want. When you later take a look at an older snapshot (let's start calling it "version"), your VCS shows you exactly how it differed from the previous one, as illustrated in figure 10.1. Example VCS's are:

- Concurrent Versions System[2] (CVS)
- Subversion[3] (SVN)
- Git

10.1 Centralized vs Decentralized Version Control

Check table 10.1 and figures 10.2, 10.2 and 10.3.

10.2 Introducing Git

Git is an open source program for tracking changes in text files. It was written by Torvald Linus; the author of the Linux operating system.

10.2.1 Git is Very Different

The first important thing to understand about Git is that it thinks about version control very differently than *Subversion* or whatever VCS tool you may be used to.

Theorem 10.1 (The Forget theorem)**.** *It is often easier to learn Git by trying to* ***forget*** *your assumptions about how version control works and try to think about it in the Git way.*

[1] Also denoted as "Revision Control Systems (RCS)", or Source Code Management (SCM) system.
[2] Very old, widespread, but not so good
[3] A modern version of CVS.

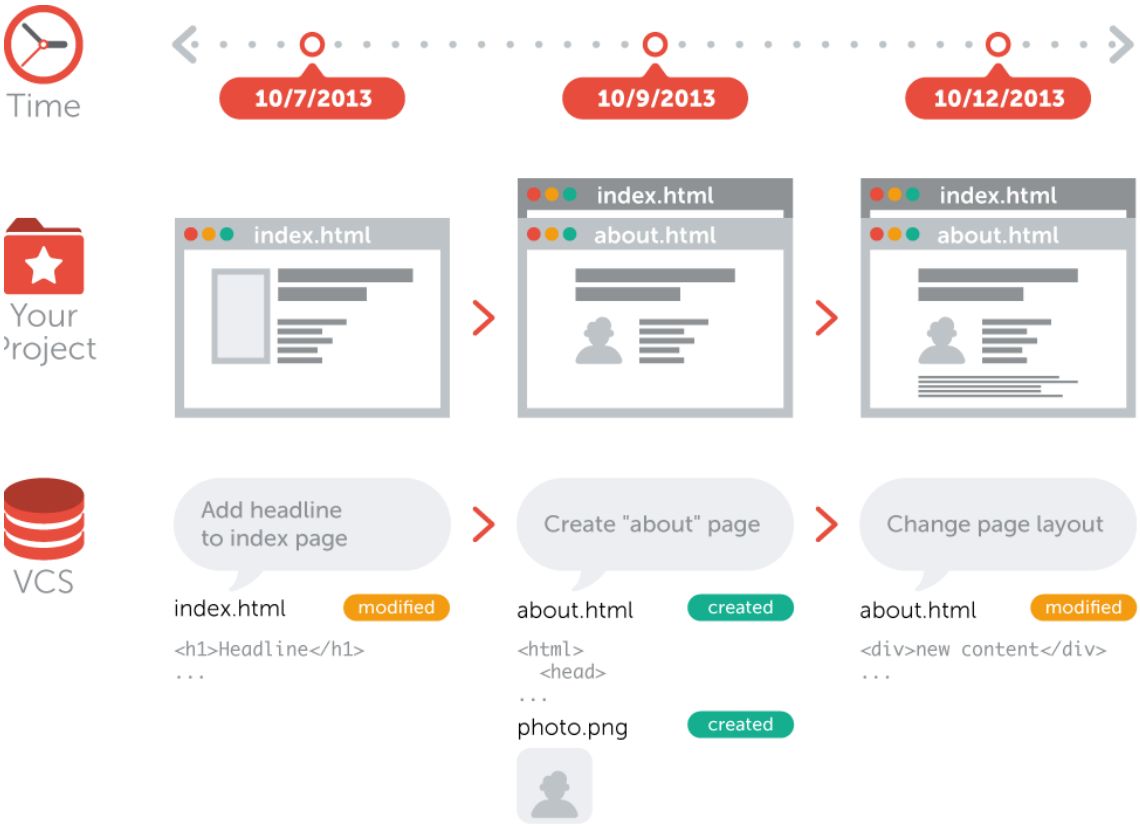


Figure 10.1: VCS illustration [1]

Centralized	Decentralized
CVS	Git
SVN	HG
...	...

Table 10.1: Typical centralized and decentralized VCS's.

10.2.2 Git GUI's

There are two main ways of working with Git:

1. Command Line Interface
   - It is recommended to learn the basics of Git on the command line first. It helps you form a deeper understanding of the underlying concepts and makes you independent from any specific GUI application.
2. GUI
   - This will make you more efficient and let you access more advanced features that would be too complex on the command line.
   - Check `https://git-scm.com/downloads/guis` for the complete list. Anyway, don't expect any GUI can replace Git commands altogether.

10.2.2.1 Tower

Tower (`www.git-tower.com`) seems to be the best GUI. Its documentation, notably [1], are concise, clear and very well written. Tower can be installed only on Mac and Windows.

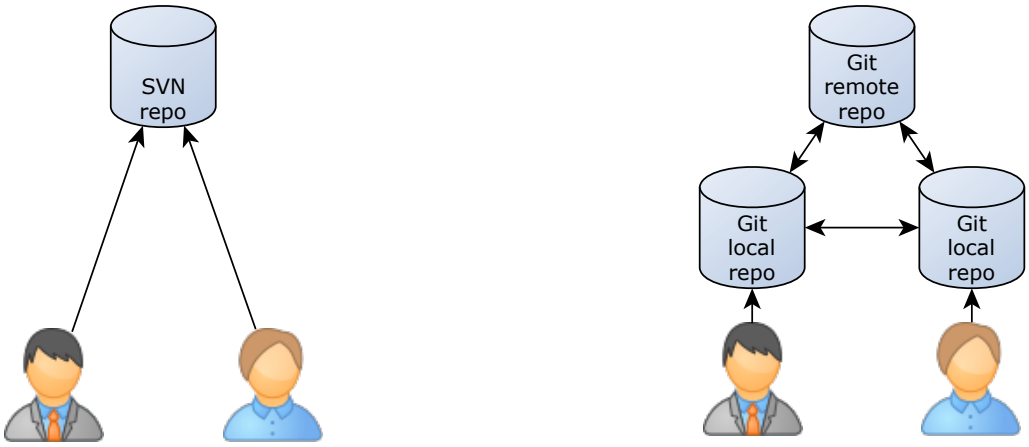


Table 10.2: Centralized versus decentralized VCS

It is however expensive and not open source.

10.2.2.2 GitKraken

GitKraken (www.gitkraken.com) on the other hand seems similar to *Tower*. It is free, cross platform but, however, not open source!

GitKraken Cheat Sheets

- GitKraken Cheat Sheet; www.gitkraken.com/resources/gitkraken-cheat-sheet
- GitKraken for GitHub Users Cheat Sheet; www.gitkraken.com/resources/gitkraken-github-cheat-sheet

Tips Using GitKraken If you use *remote* s on GitHub.com or BitBucket.org, make sure to authenticate as explained in https://support.gitkraken.com/integrations/github and https://support.gitkraken.com/integrations/bitbucket.

10.2.3 Installing Git

- Installation binaries is available for download at https://git-scm.com/downloads.
- Installation and execution guidance is available in [1, Part1; Getting Ready].

10.3 Workflow of Git

Check figures 10.4, 10.5 and 10.6.

10.3.1 Git Cheat Sheet

Check figure 10.7.

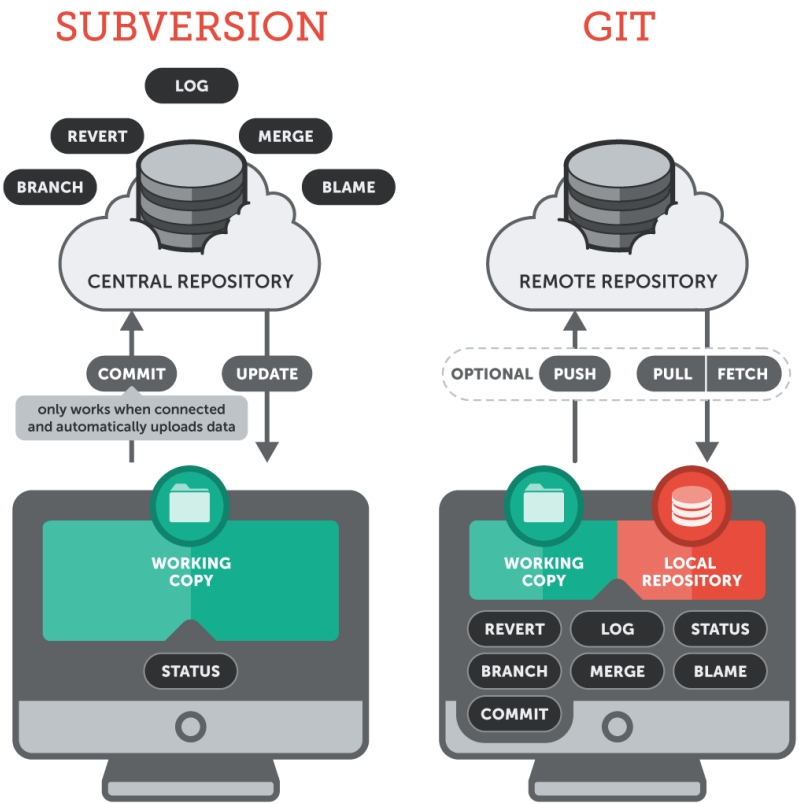


Figure 10.2: Git commands versus SVN commands [1]

10.3.2 Git Best Practices

Check figure 10.8.

10.4 Git Terminology Explained

The most important obstacle against learning Git is its awkward terminology. The following terms [9] is ordered from the most basic to the less likely to use/hear-about.

Repository consists of two things:

".git" directory is where Git stores the metadata and object database of the repository in a compressed format. It is what is copied when a repository is cloned. This directory is hidden and hence you must enable viewing hidden directories in order to see it.

working directory normally contains the contents of the *HEAD* commit, plus any local changes made.

- Reverting to older *commit* replaces the *working directory* with the snapshot of this commit.
- *checkout* a *branch* replaces the *working directory* with the snapshot of the *HEAD* *commit* of the checked-out branch.

working copy is a synonym to *working directory*

working tree is a synonym to *working directory*

staging area is generally a file in *".git" directory* that stores information about what will be included into the next *commit* . It is also called *index* .

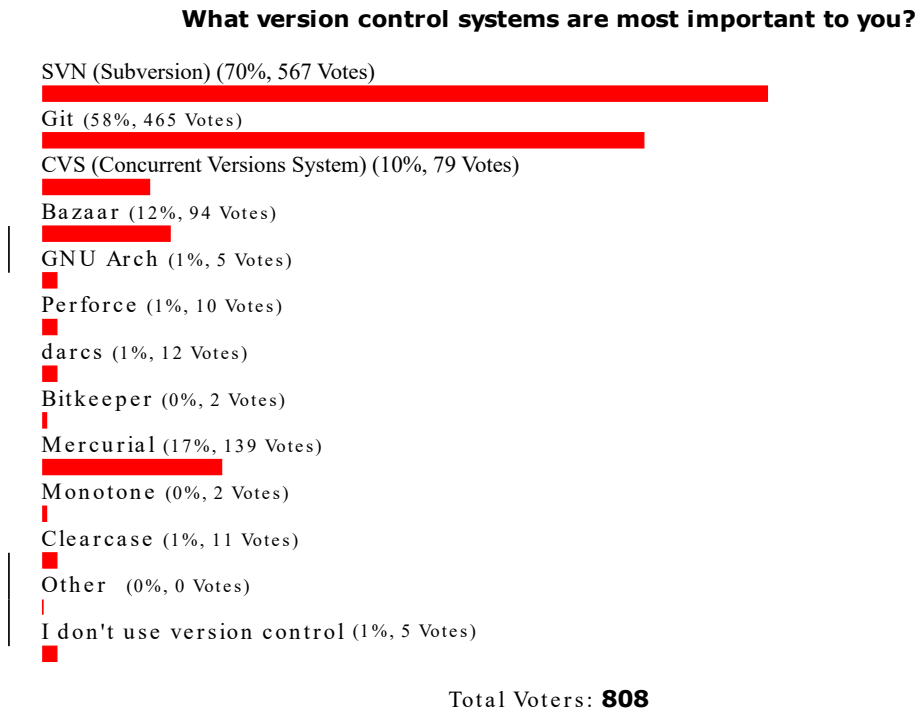


Figure 10.3: Result of a survey about favorite VCS's

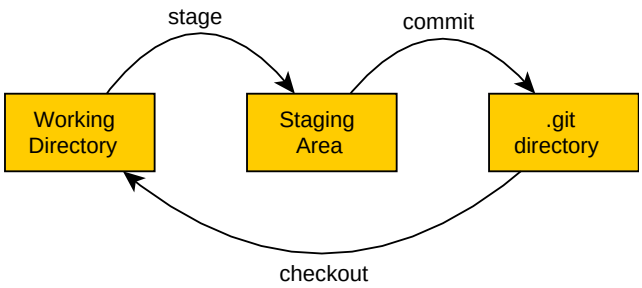
index is a synonym to *staging area*

stage adds files to the *staging area* , so that they are included in the next *commit* .

- Be warned that non-staged files may be removed (deleted) when checkingout a *branch* or reverting to older an *commit* .

add is a synonym to *stage* .

commit [1] record a snapshot of the current state of the *staging area* , marking a new version of your repository. Later on, you can revert the repository to any commit.



tag is most typically used to mark a particular *commit* .

head is a named reference to the last *commit* of a *branch* .

HEAD is a named reference to *head* of the *current branch* .

clone does the following:

1. creates a *local* copy of a *remote Repository* , including all of its *branch es*,
2. sets up tracking information[2] between each *local-remote* (upstream) *branch*
3. checkout the *local* branch corresponding to the *remote* *'s current* branch.

[1] In other other VCS's, the same thing is referred to as *revision* or *version*.

[2] This enables using `$ git push` and `$ git pull` commands without specifying further arguments identifying targeted local and *remote branch es*.

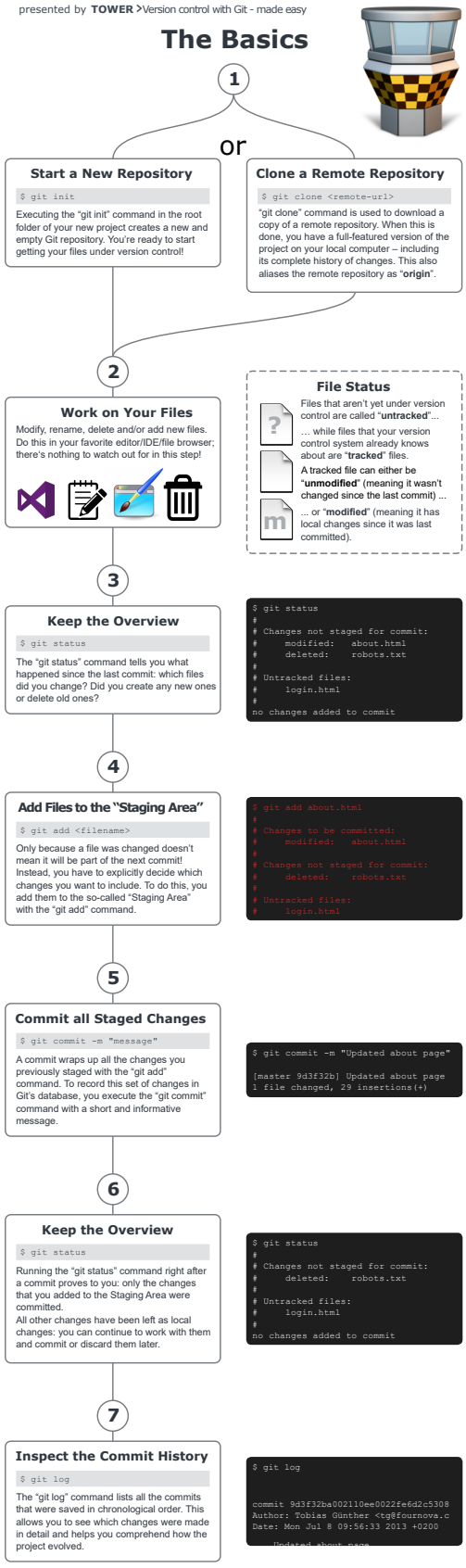


Figure 10.4: Git Basics [https://www.git-tower.com/learn/cheat-sheets/vcs-workflow with modifications]

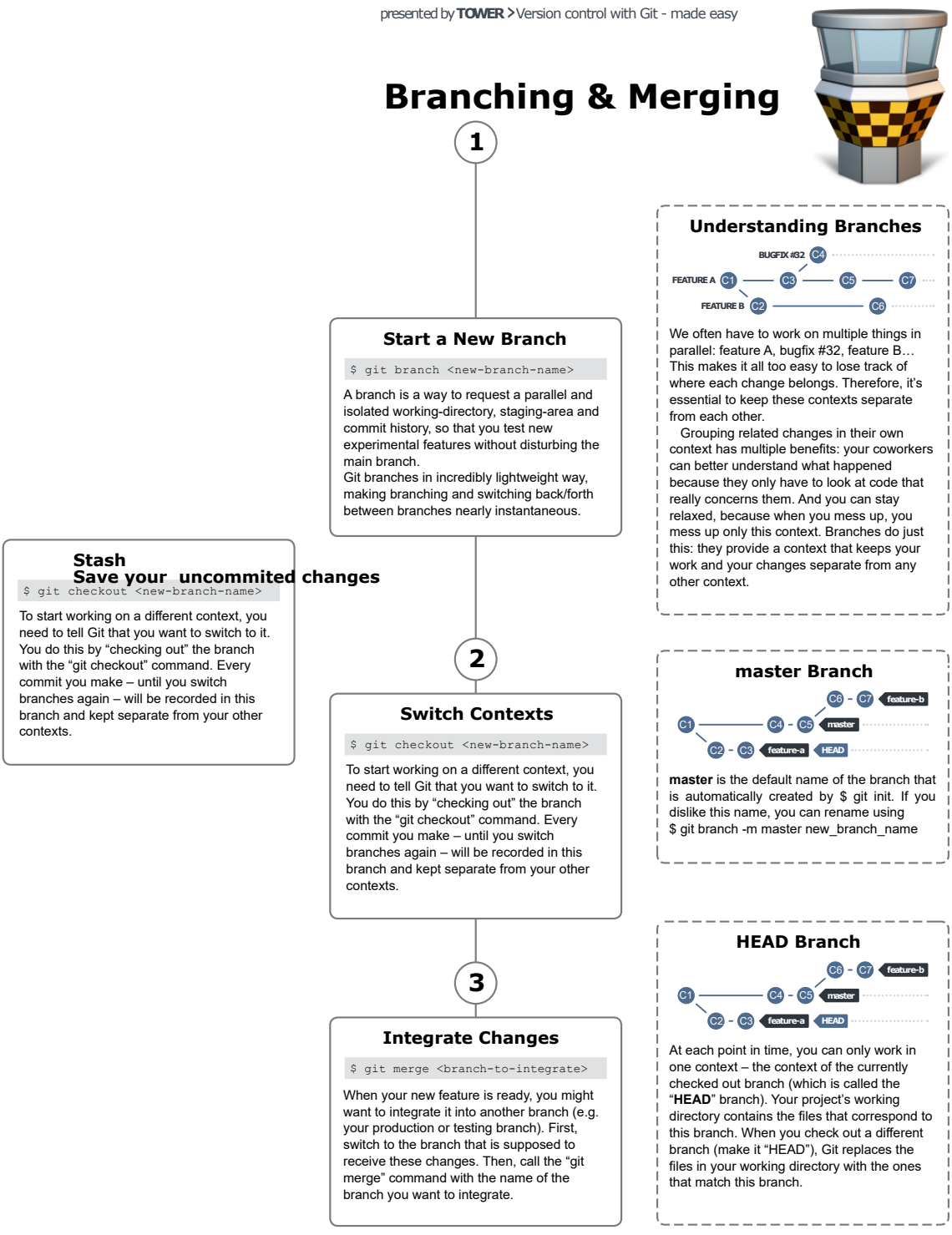


Figure 10.5: Git branching and merging [https://www.git-tower.com/learn/cheat-sheets/vcs-workflow with modifications]

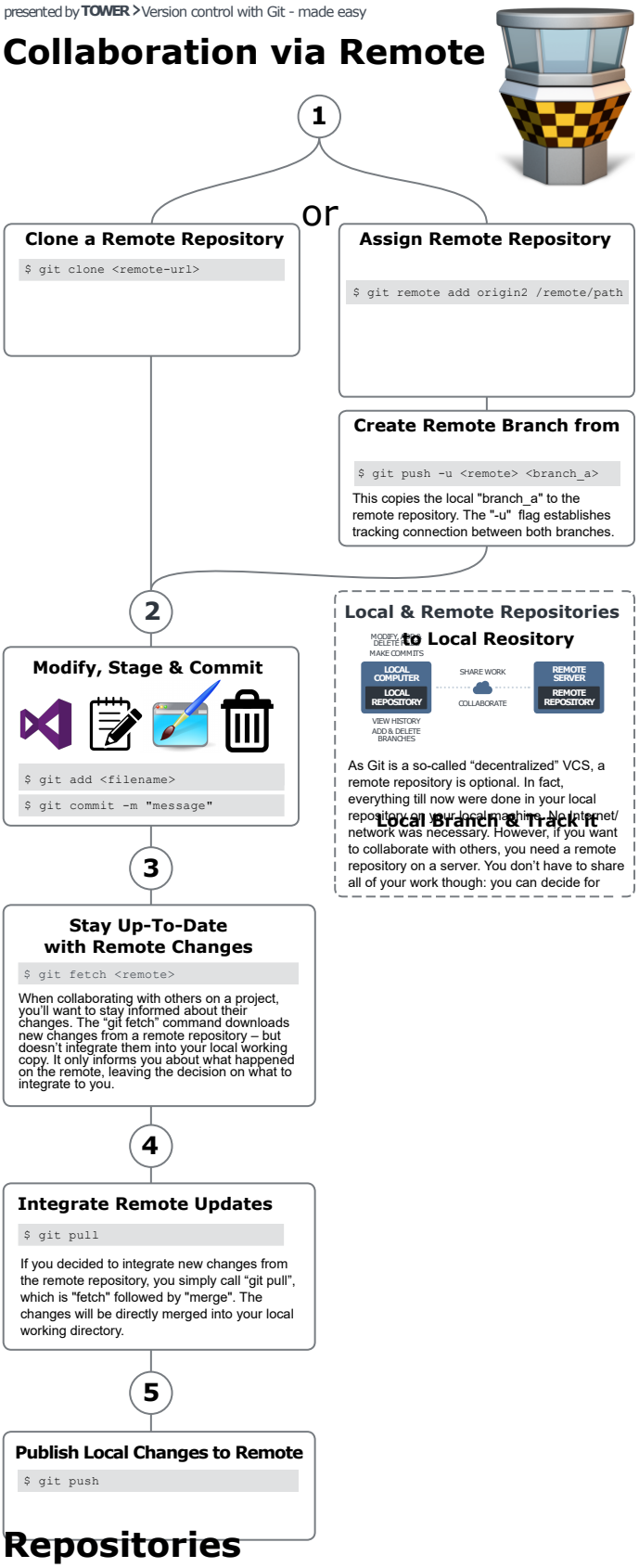


Figure 10.6: Git sharing work via *remote* repositories [https://www.git-tower.com/learn/cheat-sheets/vcs-workflow with modifications]

GITCHEAT SHEET

presented by **TOWER**>Version control with Git - made easy

CREATE

Clone an existing repository

```
$ git clone ssh://user@domain.com/repo.git
```

Create a new local repository

```
$ git init
```

LOCAL CHANGES

Changed files in your working directory

```
$ git status
```

Changes to tracked files

```
$ git diff
```

Add all current changes to the next commit

```
$ git add .
```

Add some changes in <file> to the next commit

```
$ git add -p <file>
```

Commit all local changes in tracked files

```
$ git commit -a
```

Commit staged changes

```
$ git commit
```

Change the last commit

Don't amend published commits!

```
$ git commit --amend
```

COMMIT HISTORY

Show all commits, starting with newest

```
$ git log
```

Show changes over time for a specific file

```
$ git log -p <file>
```

Who changed what and when in <file>

```
$ git blame <file>
```

BRANCHES & TAGS

List all local & remote branches

```
$ git branch -av
```

Switch HEAD branch

```
$ git checkout <branch>
```

Create a new branch based on your current HEAD

```
$ git branch <new-branch>
```

Create a new local branch tracking a remote branch, and checkout it

```
$ git checkout --track <remote/branch>
```

Delete a local branch

```
$ git branch -d <branch>
```

Mark the current commit with a tag

```
$ git tag <tag-name>
```

UPDATE & PUBLISH

List all currently configured remotes

```
$ git remote -v
```

Show information about a remote

```
$ git remote show <remote>
```

Add new remote repository, named <remote>

```
$ git remote add <shortname> <url>
```

Download all changes from <remote>, but don't integrate into HEAD

```
$ git fetch <remote>
```

Download all changes from <remote>, and merge them into HEAD

```
$ git pull <remote> <branch>
```

Publish local changes on a remote

```
$ git push <remote> <branch>
```

Delete a branch on the remote

```
$ git branch -dr <remote/branch>
```

Publish your tag s

```
$ git push --tags
```

MERGE & REBASE

Merge <branch> into your current HEAD

```
$ git merge <branch>
```

Rebase your current HEAD onto <branch>

Don't rebase published commits!

```
$ git rebase <branch>
```

Abort a rebase

```
$ git rebase --abort
```

Continue a rebase afterresolving conflicts

```
$ git rebase --continue
```

Use your configured merge tool to solve conflicts

```
$ git mergetool
```

Use your editor to manually solve conflicts and (after resolving) mark file as resolved

```
$ git add <resolved-file>
$ git rm <resolved-file>
```

UNDO

Discard all local changes in your working directory

```
$ git reset --hard HEAD
```

Discard local changes in aspecific file

```
$ git checkout HEAD <file>
```

Revert a commit (by producing a new commit with contrary changes)

```
$ git revert <commit>
```

Reset your HEAD pointer to a previous commit …and discard all changes since then

```
$ git reset --hard <commit>
```

…and preserve all changes as unstaged changes

```
$ git reset <commit>
```

…and preserve uncommitted local changes

```
$ git reset --keep <commit>
```

Figure 10.7: Git Cheat Sheet [https://www.git-tower.com/learn/cheat-sheets/git]

VERSION CONTROL

BEST PRACTICES

COMMIT RELATED CHANGES

A commit should be a wrapper for related changes. For example, fxing two diferent bugs should produce two separate commits. Small commits make it easier for other developers to understand the changes and roll them back if something went wrong.
With tools like the staging area and the ability to stage only parts of a fle, Git makes it easy to create very granular commits.

TEST CODE BEFORE YOU COMMIT

Resist the temptation to commit something that you «think» is completed. Test it thoroughly to make sure it really is completed and has no side efects (as far as one can tell). While committing half-baked things in your local repository only requires you to forgive yourself, having your code tested is even more important when it comes to pushing/sharing your code with others.

USE BRANCHES

Branching is one of Git's most powerful features - and this is not by accident: quick and easy branching was a central requirement from day one. Branches are the perfect tool to help you avoid mixing up diferent lines of development. You should use branches extensively in your development workflows: for new features, bug fxes, ideas...

COMMIT OFTEN

Committing often keeps your commits small and, again, helps you commit only related changes. Moreover, it allows you to share your code more frequently with others. That way it's easier for everyone to integrate changes regularly and avoid having merge conficts. Having few large commits and sharing them rarely, in contrast, makes it hard to solve conficts.

WRITE GOOD COMMIT MESSAGES

Begin your message with a short summary of your changes (up to 50 characters as a guideline). Separate it from the following body by including a blank line. The body of your message should provide detailed answers to the following questions:
› What was the motivation for the change?
› How does it difer from the previous implementation?
Use the imperative, present tense («change», not «changed» or «changes») to be consistent with generated messages from commands like git merge.

AGREE ON A WORKFLOW

Git lets you pick from a lot of diferent workflows: long-running branches, topic branches, merge or rebase, git-flow... Which one you choose depends on a couple of factors: your project, your overall development and deployment workflows and (maybe most importantly) on your and your teammates' personal preferences. However you choose to work, just make sure to agree on a common workflow that everyone follows.

DON'T COMMIT HALF-DONE WORK

You should only commit code when it's completed. This doesn't mean you have to complete a whole, large feature before committing. Quite the contrary: split the feature's implementation into logical chunks and remember to commit early and often. But don't commit just to have something in the repository before leaving the ofce at the end of the day. If you're tempted to commit just because you need a clean working copy (to check out a branch, pull in changes, etc.) consider using Git's «Stash» feature instead.

VERSION CONTROL IS NOT A BACKUP SYSTEM

Having your fles backed up on a remote server is a nice side efect of having a version control system. But you should not use your VCS like it was a backup system. When doing version control, you should pay attention to committing semantically (see «related changes») - you shouldn't just cram in fles.

HELP & DOCUMENTATION

Get help on the command line

```
$ git help <command>
```

FREE ONLINE RESOURCES

http://www.git-tower.com/learn
http://rogerdudler.github.io/git-guide/
http://www.git-scm.org/

Figure 10.8: Git Best practices [https://www.git-tower.com/learn/cheat-sheets/git]

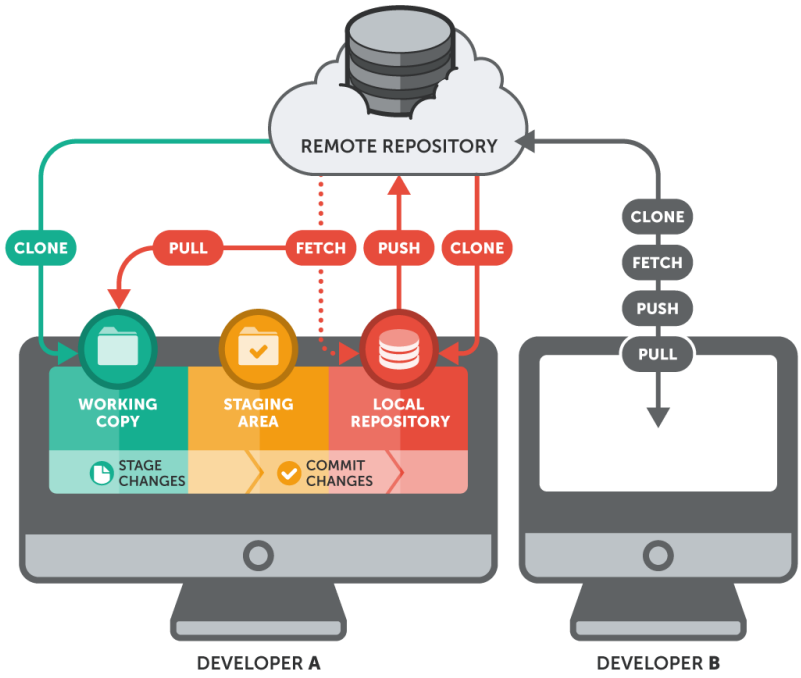


Figure 10.9: Basic *remote* workflow [1]

- `$ git clone <remote-url>` automatically aliases *remote* *Repository* as *origin* .
- If you prefer another alias for the *remote* *Repository* , clone using `$ git clone -o remote_alias <remote-url>`

remote is a repository that is used to track the *local* repository but resides somewhere else. Teams are using *remote* repositories to share & exchange data: they serve as a common base where everybody can publish their own changes and receive changes from their teammates.

- Remote repository can be usual or *bare* repository.
- Remote data can be updated/synced using with/from *local* repository using *fetch* , *pull* and *push* , as illustrated in figure 10.9.

- A *local repository* can have several *remotes*.

upstream refers to the *remote* with which the *local* syncs.

downstream refers to the *local*, as compared to *upstream* .

origin is the default name assigned to *remote* by `$ git clone`. If you dislike this name, you can rename using `$ git remote rename origin new_origin_name`

pull updates the *current local branch* , and hence the *working directory* , with the *upstream* *branch* modifications. *pull* can work in any of the following ways. Differences between them are depicted in figure 10.10.

- `$ git pull` performs two operations: (1) *fetch* the *upstream* *branch* updates and (2) *merge* them into the *current local* branch.
- `$ git pull --rebase` performs two operations: (1) *fetch* the *upstream branch* updates and (2) *rebase* the latest *local commit* on top of the *upstream branch* . This is suitable for updating after a short time.

push uploads all the new *commit* *s* from the *current local branch* to the corresponding *upstream* branch. If the *upstream branch* was a **direct** ancestor to the *local branch*,

push completes. Otherwise, the push is rejected. In this case, you have to *pull* the *upstream branch* first before you are can push.

- If the owner of the *local* repository does not have permission to *push* to *remote*, then *push*ing *local* to *remote* is not possible. In this case instead, the owner of the *local* repository sends a *pull request* to the owner of the *remote* repository.

pull request is a request from the owner of a *local* repository to the owner of the *remote* repository to pull his changes. *remote*'s owner can use *diff* to review the changes and may selectively accepted/rejected changes.

- If the owner of the *local* repository has permission to *push* to *remote*, he can instead directly *push* *local* to *remote*.
- *pull request* is an announcing method, and are not a feature of Git. So it depends on the hosting website[1] and has no Git command.

fetch fetches *branch es* from a *remote Repository*, along with the objects necessary to complete their histories.

- Fetch will not touch any of your *local branch es* or your *working directory*. It just downloads data from the specified *remote* and makes them visible so that you can decide if you want to integrate new changes into your *local Repository*.

diff is a utility software that calculates and displays the differences between two files. Typically in Git, *diff* is used to determine the differences between two committed versions of a file. In fact, the word *diff* became a generic term for the utility software and its output result as well.

branch is a way to request a parallel and isolated *working directory*, *staging area* and *commit* history, so that you test new experimental features without disturbing the main branch[2]. A local branch that you create on your machine is kept private to you until you explicitly decide to publish it using *push*. This means that it's perfectly possible to keep some of your work private while sharing only certain other branches with the world.

checkout a *branch* means to switch to this branch, ***replace***[3] the *working directory* with the snapshot of the *head* of this branch and update the *staging area* and *HEAD* to point to this branch.

master is the default name of the branch that is automatically created by `$ git init`. If you dislike this name, you can rename using `$ git branch -m master new_branch_name`

merge tries to merge a *branch* into the *current local branch*. As a result, it creates a merge *commit* combining both *branch es*.

- *merge* modifies the *current working directory*. Therefore, *stash* your uncommitted modifications before *merge*.
- *merge* integrates a *branch*; not individual *commit s*

[1] Such as GitHub.com and BitBucket.org

[2] In many VCS tools, *branching* a somewhat expensive process, often requiring creating a new copy of the source code directory, which can take a long time for large projects. Therefore, Git's branching model is referred to as a "*killer feature*" that sets superior in the VCS community. This is because Git branches in incredibly lightweight way, making branching nearly instantaneous, and switching back and forth between branches generally just as fast. Unlike many other VCSs, Git encourages workflows that *branch* and *merge* often, even multiple times in a day.

[3] Non *commit*ted, *stage*d or *stash*ed files may be removed (deleted). Therefore, it is advisable to *commit*/*stage*/*stash* your modifications before checkingout.

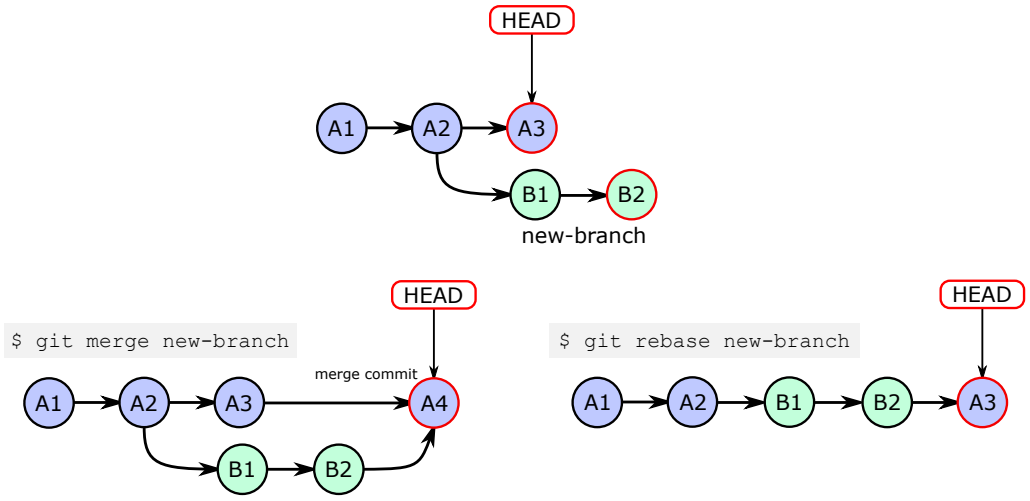


Figure 10.10: *merge* versus *rebase*

- If the *branch* to be merged happens to be descendant of *HEAD*, *merge* will automatically perform *fast forward* *merge*, as depicted in figure 10.11.
- If the merged *branch* *es* changed the same lines in that same file, or if one deleted it while the other modified it, Git simply cannot know what is correct. Git will then mark the file as having a ***conflict*** - which you'll have to solve before you can continue your work. Details are in [1, Part 4; Dealing With Merge Conflicts].
- You can always undo *merge* and go back to the state before a *conflict* occurred.
- Merge *conflicts* can only occur on *local branch* - and not on *remote*. That is, Merge *conflicts* will never bring the complete team to a halt.

rebase reapply a series of changes from a branch to a different base, as depicted in figure 10.10.

Caution: *rebase* rewrites history. Therefore, *rebase* should only be used for cleaning up local commits. **Do not** *rebase* commits that have already been published to *remote*.

fast forward is a special type of *merge* that can occur while merging a *branch* that happens to be descendant of *HEAD*. In this case, *merge* will not make a new *merge commit*. Instead, *merge* will *rebase* the merged branch on top of *HEAD*, as depicted in figure 10.11. This is simply performed in this case just by pointing *HEAD* to the *head* of the merged branch. For more information, refer to [1, Part 4; Rebase as an Alternative to Merge].

stash saves your *working directory* modifications away and reverts the *working directory* to match the *HEAD* *commit*. This is needed in case your modifications are not yet

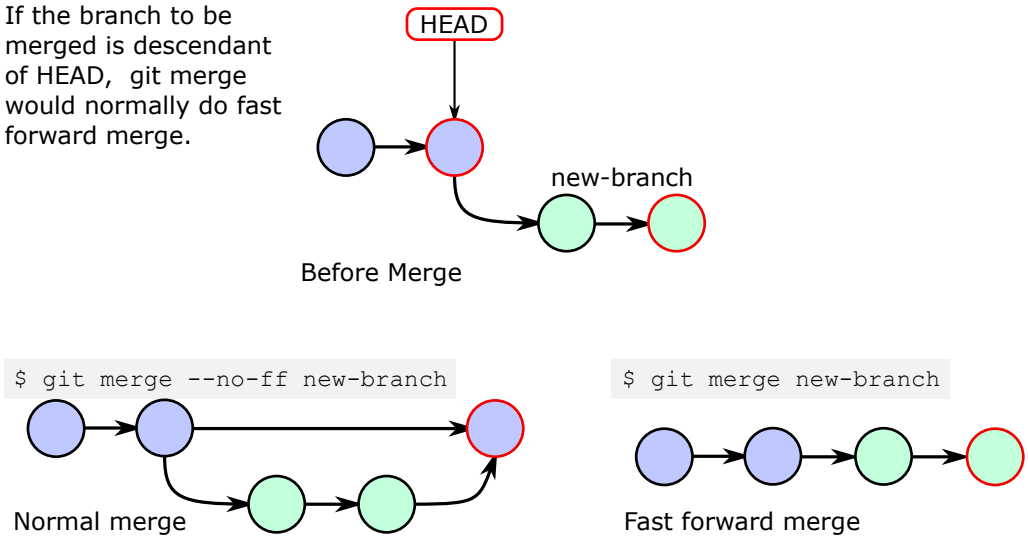


Figure 10.11: *fast forward* *merge*

ready for a *commit* , while you are interrupted with another job that will overwrite the *working directory* using for example:

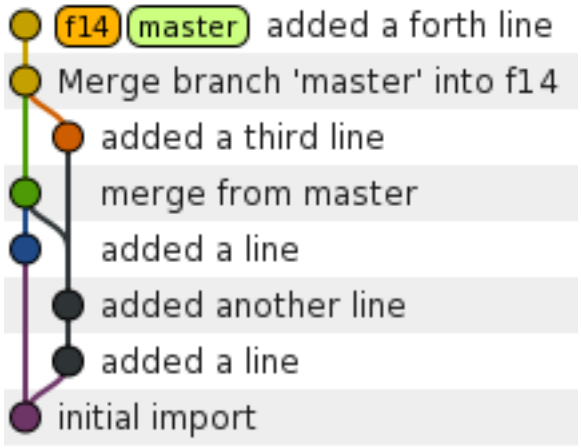
- *checkout*
- *merge*
- *rebase*
- *pull*

In this case, *stash* can do the trick by saving your modifications and resetting the *working directory* so that you can safely start a new side job. After finishing the side job, you can restore (*pop*) your stashed work and continue updating it.

pop restores stashed modifications on top of the *working directory* , i.e., do the inverse *stash* .

cherry pick means to extract changes introduced by a commit in some branch, and apply/repeat them on the tip of the current branch as a new commit.

DAG Directed acyclic graph.



resolve is fixing up manually what a failed automatic *merge* left behind.

blame describes the last modification to each line of a file, which generally displays the version, author and time. This is helpful, for example, in tracking down when a feature was added, or which commit led to a particular bug.

Fork fork is a copy of another repository that lives on your account. Forks allow you to freely make changes to a project without affecting the original. Forks remain attached to the original, allowing you to submit a pull request to the original's author to update with your changes. You can also keep your fork up to date by pulling in updates from the original.

SHA-1 (Secure Hash Algorithm 1) a cryptographic hash function used as a synonym for object name.

submodule is a repository inside another repository (the latter of which is called *superproject*).

superproject is a repository that references repositories of other projects in its working tree as *submodule* . The superproject knows about the names of (but does not hold copies of) commit objects of the contained submodules.

Hook is a script that runs automatically every time a particular event occurs in a Git repository. Hooks let you customize Git's internal behavior and trigger customizable actions at key points in the development life cycle.

prune removes unreachable objects.

bare repository is intended to be solely used as a *remote* repository. That is, it is not used for working on files, but rather for sharing and exchanging code between developers. Hence, a bare repository contains no *working directory* and stores git version history in the root folder of the repository instead of in a *".git" directory* . Customarily, bare repositories are given a ".git" extension. A blank bare repository can be created with `$ git init --bare`. Alternatively, it can be cloned from a local repository with `$ git clone --bare`.

squash Squashing a commit is to take the changes from one commit and add them to its parent commit.

10.5 Undoing Things

10.5.1 Revert vs reset

Whereas reverting is designed to safely undo a public commit, git reset is designed to undo local changes. Because of their distinct goals, the two commands are implemented differently: resetting completely removes a changeset, whereas reverting maintains the original changeset and uses a new commit to apply the undo.

10.6 Merge Conflicts

[Git Tower ebook]

Git was nice enough to mark the problematic area in the file by enclosing it in "<<<<<<< HEAD" and ">>>>>>> [other/branch/name]".

- Consequently, don't use external tool to merge non-text-designed documents such as LyX or others.

10.7 Diff of Complex Text Files of non-text files

One essential operation of version control is to identify the differences between different versions of a file. By default this is performed by the *diff* tool.

While *diff* is pretty good for plain text files, it is much less useful in the case of LyX files, which have more a complicated structure.

10.8 Further Details

10.8.1 Excluding Files from Version Control

Check [1, Part 1; Starting with an Unversioned Project; Ignoring Files].

10.8.2 Submodules

Check [1, Part 4; Submodules].

10.8.3 Undoing Things

Check [1, Part 4; Undoing Things].

10.8.4 Restore a Previous Version

Check https://www.git-tower.com/learn/git/faq/restore-repo-to-previous-revision.

Appendix A

Matlab Codes

Code A.1: SDOF_Free_Response_Visc_main

```
function SDOF_Free_Response_Visc_main()
clc
close all

set(groot,'DefaultAxesColorOrder',[0,0,1;0,0,0;1,0,0;0,0.5,0;1,0,1])
set(groot,'DefaultAxesLineStyleOrder','-|--|-.')
set(groot,'DefaultLineLineWidth',1);
set(groot,'DefaultAxesFontName','Times')

w_n=1;
x0=-1;
v0=0;

zeta_vec=[0,.1,.2,.4,1/sqrt(2),1,2];
legend_string={'$\zeta = 0$','$\zeta = 0.1$','$\zeta = 0.2$','$\zeta = 0.4$','$\zeta=1/\sqrt{2}$','$\zeta = 1$','$\zeta = 2$'};

t_vec=linspace(0,4*pi,500);

figure
hold on
for n=1:length(zeta_vec)
    x_vec=SDOF_Free_Response_Visc(w_n,zeta_vec(n),x0,v0,t_vec);
    plot(w_n*t_vec,x_vec)
end

title('$x(t)$ for $\omega_{n}=1$, $x_{0}=-1$ and $\dot{x}_{0}=0$','interpreter','latex');
xlabel('$\omega_{n} t$','interpreter','latex');
legend(legend_string,'interpreter','latex','Location','SouthEast');

grid on
ax=gca;
ax.XTick=0:pi:4*pi;
```

```
ax.XTickLabel={'0','\pi','2\pi','3\pi','4\pi'};
ax.XAxis.MinorTickValues=setdiff(0:pi/2:4*pi,0:pi:4*pi);
ax.XMinorGrid='on';
ax.XLim=[0,4*pi];

set(groot,'DefaultAxesColorOrder','remove')
set(groot,'DefaultAxesLineStyleOrder','remove')
set(groot,'DefaultLineLineWidth','remove');
set(groot,'DefaultAxesFontName','remove')

export_figure(gcf,'',{'SDOF_FreeResponse'})
```

Code A.2: function SDOF_Free_Response_Visc.m

```
function x_vec=SDOF_Free_Response_Visc(w_n, zeta, x0, x_dot_0, t_vec)

if zeta~=1
    w_d=w_n*sqrt(1-zeta^2);
    x_vec=exp(-zeta*w_n*t_vec).*(x0*cos(w_d*t_vec)+(zeta*w_n*x0+
       x_dot_0)*sin(w_d*t_vec)/w_d);
else
    x_vec=exp(-w_n*t_vec).*(x0+(w_n*x0+x_dot_0)*t_vec);
end
```

Code A.3: function export_figure

```
function export_figure(fig_handle_vec, ...
                  Expand,filenames,resolution,pictureFormat_cVec,
                     dimScale) %Optional arguments

if nargin<2
    Expand='';
end

if nargin<4
    resolution=600;
elseif isempty(resolution)
    resolution=600;
end

if nargin<5
    pictureFormat_cVec={'pdf'};
elseif isempty(pictureFormat_cVec)
    pictureFormat_cVec={'pdf'};
else
    if ~iscell(pictureFormat_cVec)
        error('pictureFormat must be cell array of strings.')
    end
```

```
end

if nargin<6
    dimScale=[];
end

printFlag_cVec=cell(size(pictureFormat_cVec));
for n=1:length(pictureFormat_cVec)
    if strcmpi(pictureFormat_cVec{n},'emf')
        if ispc
            printFlag_cVec{n}='meta';
        else
            error('Matlab␣cannot␣export␣emf␣except␣under␣Windows.');
        end
    else
        printFlag_cVec{n}=lower(pictureFormat_cVec{n});
    end
end

if min(size(fig_handle_vec,1),size(fig_handle_vec,2))~=1
    error('h␣must␣be␣1␣D␣vector'),
end

if ~iscellstr(filenames)
    error('filenames␣must␣be␣a␣cell␣string␣of␣the␣same␣length␣as␣
       h_vec');
end

if nargin>2
    if length(fig_handle_vec)~=length(filenames)
        error('h␣&␣filenames␣must␣be␣of␣the␣same␣length');
    end
end

if ~isempty(Expand)
    if ischar(Expand)
        if (~strcmpi(Expand,'||') && ~strcmpi(Expand,'=='))
            error('you␣must␣input␣''||''␣or␣''==''')
        end
    end
end

for i=1:length(fig_handle_vec)
    f_OriginalUnit=get(fig_handle_vec(i),'Units');
    set(fig_handle_vec(i),'papertype','A4');
    if ~isempty(Expand)
        if ischar(Expand)
            if strcmpi(Expand(1:2),'||')
```

```
                set(fig_handle_vec(i), 'PaperOrientation', 'portrait'
                    );
            elseif strcmpi(Expand(1:2),'==')
                set(fig_handle_vec(i), 'PaperOrientation', 'landscape')
                    ;
            end
        end

        if ischar(Expand)
            if strcmpi(Expand,'||') || strcmpi(Expand,'==')
                a=get(fig_handle_vec(i),'papersize');
                set(fig_handle_vec(i), 'PaperPositionMode', 'manual');
                set(fig_handle_vec(i),'PaperPosition',[0 0 a(1) a(2)])
                    ;
                set(fig_handle_vec(i),'Units',get(fig_handle_vec(i),'
                    PaperUnits'));
                set(fig_handle_vec(i),'Position',[0 0 a(1) a(2)]);
                set(fig_handle_vec(i),'Units',f_OriginalUnit);
                set(0,'CurrentFigure',fig_handle_vec(i)),
                drawnow
            else
                set(fig_handle_vec(i), 'PaperPositionMode', 'auto');
            end
        end
        if ~isempty(dimScale)
            pos=get(fig_handle_vec(i),'PaperPosition');
            set(fig_handle_vec(i), 'PaperPositionMode', 'manual');
            set(fig_handle_vec(i),'PaperPosition',[pos(1:2),pos(3:4).*
                dimScale/max(dimScale)]);
        end
    end
end

for i=1:length(fig_handle_vec)
    for n=1:length(printFlag_cVec)
        if any(strcmp(printFlag_cVec{n},{'emf','pdf','eps','epsc','
            svg'}))
            renderer='-painters';
        elseif any(strcmp(printFlag_cVec{n},{'png','jpg'}))
            renderer='-opengl';
        end
        if nargin<3
           print(['-r',int2str(resolution)], renderer, ['-d',
              printFlag_cVec{n}],['-f',int2str(double(fig_handle_vec(
              i)))]);
        else
           print(['-r',int2str(resolution)], renderer, ['-d',
              printFlag_cVec{n}],['-f',int2str(double(fig_handle_vec(
```

```
            i)))],[filenames{i},'.',pictureFormat_cVec{n}]);
        end
    end
end

%If "strawberry perl" and Miketex is installed
if nargin>=3
    temp_env=getenv('LD_LIBRARY_PATH');
    setenv('LD_LIBRARY_PATH', '')

    if any(strcmpi(pictureFormat_cVec,'pdf'))
        [status,~]=system('where␣pdfcrop');
        if status,warning('pdfcrop␣is␣not␣installed.␣Please␣install␣
           it␣through␣TeXLive␣or␣MiKTeX.'),end
    end

    if any(strcmpi(pictureFormat_cVec,'png')) || any(strcmpi(
       pictureFormat_cVec,'jpg'))
        if ispc
            [status,~]=system('where␣magick');
            if status,warning('Imagemagick␣is␣not␣installed.'),end
        else
            [status,~]=system('where␣convert');
            if status,warning('Imagemagick␣is␣not␣installed.'),end
        end
    end

    for n=1:length(pictureFormat_cVec)
        if strcmpi(pictureFormat_cVec{n},'pdf')
            for i=1:length(fig_handle_vec)
               system(['pdfcrop␣"',filenames{i},'.pdf"␣"',filenames{i
                  },'.pdf"']);
            end
        elseif any(strcmpi(pictureFormat_cVec{n},{'png','jpg'}))
            for i=1:length(fig_handle_vec)
                system(['magick␣convert␣"',filenames{i},'.',
                   pictureFormat_cVec{n},'"␣-trim␣"',filenames{i},'.',
                   pictureFormat_cVec{n},'"']);
            end
        end
    end
    setenv('LD_LIBRARY_PATH', temp_env)
end
```

This page is intentionally left blank!

References

[1] T. Günther, *LEARN VERSION CONTROL WITH GIT – A step-by-step course for the complete beginner*, A. Rinaß, Ed., command line version. [Online]. Available: www.git-tower.com/learn/git/ebook/en/command-line/introduction

[2] The LyX Team, *LyX's detailed Figure, Table, Floats, Notes, Boxes and External Material manual*, 2nd ed., accessible from LyX's help menu as "Embedded Objects".

[3] ——, *The LyX User's Guide*, 2nd ed., accessible from LyX's help menu.

[4] ——, *LyX's detailed Math manual*, 2nd ed., accessible from LyX's help menu as "Math".

[5] Wikipedia, "Reference management software — wikipedia, the free encyclopedia," 2016, [Online; accessed 7-October-2016]. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Reference_management_software&oldid=743035115

[6] E. Francese, "Usage of reference management software at the university of torino," vol. 1, no. 4, 2013. [Online]. Available: http://leo.cineca.it/index.php/jlis/article/view/8679

[7] Wikipedia, "Comparison of reference management software — wikipedia, the free encyclopedia," 2016, [Online; accessed 17-November-2016]. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Comparison_of_reference_management_software&oldid=749999200

[8] T. Bah, *Inkscape: Guide to a Vector Drawing Program*, 4th ed. Pearson Education, 2011.

[9] "gitglossary manual page." [Online]. Available: www.kernel.org/pub/software/scm/git/docs/gitglossary.html

[10] B. Lynn, *Git Magic*. CreateSpace Independent Publishing Platform, 2010. [Online]. Available: http://www-cs-students.stanford.edu/~blynn/gitmagic/

[11] "Git reference." [Online]. Available: http://gitref.org/

[12] "Github glossary." [Online]. Available: https://help.github.com/articles/github-glossary/

[13] "git - the simple guide." [Online]. Available: http://rogerdudler.github.io/git-guide/

This page is intentionally left blank!

Index

A
add, 39
Adobe Illustrator, 21

B
bare, 49
blame, 48
bmp, 21
branch, 46

C
checkout, 46
cherry pick, 48
clone, 39
commit, 39
Corel Draw, 21
CVS, 35

D
diff, 46
downstream, 45

E
emf, 21
eps, 21

F
fast forward, 47
fetch, 46
Fork, 49
Function plotter, 27

G
Git, 35

H
HEAD, 39
head, 39
hook, 49

I
IDE, 3
index, 39
Inkscape, 21, 25

J
jpg, 21

L
LaTeX, 3
LyX, 7

M
master, 46
merge, 46
MiKTeX, 6

O
origin, 45

P
pdf, 21
png, 21
pop, 48
prune, 49
pull, 45
pull request, 46
push, 45

R
Raster graphics, 21
remote, 45
Repository, 38

S
SHA-1, 49
squash, 49
stage, 39
stash, 47
svg, 21
SVN, 35

T
tag, 39
Tex Live, 6
TexText, 27

U
upstream, 45

V
Vector graphics, 21

W
working copy, 38
working directory, 38
working tree, 38