Minicurso de LaTeX

Viviane Maranhão
viviane@ime.usp.br

Março de 2013

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

LaTeX Básico

1.1 Introdução

Este é um tutorial bem simples de LaTeX. Existe muito material disponível na internet, na documentação dos pacotes utilizados e também nas referências no final desse texto.

O LaTeX é um sistema de preparação de documentos de alta qualidade tipográfica, muito indicado para produção de documentos e textos científicos. Também pode ser utilizado para produção de outros documentos como relatórios, livros, apresentações, listas de exercícios e apostilas.

Um documento em LaTeX pode ser escrito em qualquer editor de texto puro, uma vez que deverá passar por um processo de compilação para produzir o resultado desejado. Uma das vantagens de utiliza-lo é que não é necessario se preocupar com formatação, numeração de páginas e capítulos entre outras questões estruturais do documento pois isto é feito automaticamente.

1.1.1 Utilizando LaTeX nos sistemas operacionais

Existem distribuições de LaTeX para diversos sistemas operacionais. Alguns editores de texto específicos para LaTeX possuem comandos prontos para o usuário, algum tipo de *help* embutido sobre comandos, recurosos de autocompletar e chamam diretamente o compilador, o que pode facilitar o trabalho, mas essencialmente, são editores de texto puro. Alguns exemplos de distribuições e editores:

- **Para Windows:** Distribuição MikTeX conjuntamente com o editor de textos Led
- **Para Linux:** TeX Live conjuntamente com o editor de textos Kile

Essencialmente, para criar um documento .pdf a partir de um arquivo fonte `arquivo.tex` executamos
`pdflatex arquivo.tex`
Eventualmente mais de uma vez para que as referências sejam criadas corretamente.

1.2 Estrutura de um documento

Um documento em LaTeX é composto de um *preâmbulo* e um *corpo*. No preâmbulo são feitas as declarações de tipo de documento, configurações globais e são carregados os pacotes. O corpo contém o texto propriamente dito.

1.2.1 Preâmbulo

Todo documento em LaTeX deve começar com a declaração de que classe de documento está usando através do comando

`\documentclass`[*opções*]{*classe*}

As opções referem-se às opções do documento como: tamanho do texto (10pt, 12pt, 14pt), formato do papel (a4paper, a5paper, b5paper, letterpaper, legalpaper, executivepaper), o número de colunas (onecolumn, twocolumn), orientação da folha (landscape), impressão só frente ou frente e verso (oneside, twoside), se deve iniciar uma nova página após o título do documento ou não (titlepage, notitlepage), se deve iniciar um novo capítulo em qualquer página ou somente nas paginas da direita (openright, openany), entre outras. Podemos passar mais de uma opção separando-as por vírgula. e nem todas as opções funcionam para todas as classes de documento. A classe é referente à classe do documento, pode ser: article, report, letter, book, slides, ou alguma outra vinda de instalação de um pacote externo, e deve ser única.

Neste documento utilizamos`\documentclass`[12pt]{book}.

Este também é o local para a declaração dos pacotes utilizados através do comando

`\usepackage`[*opções*]{*pacote*}

As opções variam de acordo com o pacote utilizado. Alguns pacotes (e suas opções) que podem ser úteis são:

- **babel**: Pacote para hifenização automática do texto. Para usar português do Brasil passar `brazil` como opção.
- **inputenc**: Especifica o tipo de encoding utilizado no fonte documento, pode ser cp1252, utf8, latin1, entre outros. Com este pacote caracteres com acentos podem ser digitados diretamente no código fonte, sem a necessidade de utilizar comandos especiais.
- **fontenc**: Caracteres com acentos são considerados um bloco só no arquivo gerado quando `T1` é passado como opção, o que faz com que a hifenização funcione em palavras acentuadas e é útil inclusive para usar comandos de copiar e colar no documento pronto.
- **indentfirst**: Adiciona identação (tab) do primeiro parágrafo de cada capítulo, seção ou subseção.
- **amssymb, amsmath**: Habilitam alguns caracteres e comandos matemáticos especiais.
- **hyperref**: Para utilizar links internos e externos no documento. Como opções é possivel especificar a cor de cada tipo de link, entre outras.
- **hypcap**: Este é um pacote para ser usado em conjunto com o `hyperref` para fazer com que o link interno leve para o início de uma figura ou tqabela e não para o fim, onde está a legenda. A opção `all` habilite o pacote para figuras e tabelas, numeradas ou não.
- **graphicx**: Para inserir figuras em png. Como opção passamos `pdftex` para especificar que estamos produzindo um documento em .pdf.
- **fancyhdr**: Para personalizar o cabeçalho e rodapé.
- **abntex**: Pacote para escrever nas normas da abnt.
- **natbib**: Pacote para modificar a apresentação da bibliografia referenciada no texto.
- **geometry**: Pacote para configurar as dimensões das margens e largura da folha.

- **tocbibind**: Permite que se acrescente a bibliografia, índices e glorrarios no sumário.

1.2.2 Corpo

O corpo do documento consiste em tudo aquilo que está inserido entre os seguintes comandos que delimitam o início e o final do documento:

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

Todos os capítulos, seções e subseções devem estar inseridos entre estes comandos, o que for escrito após o fim do documento é ignorado pelo compilador.

1.3 Comandos de texto

O arquivo fonte do LaTeX contém o texto a ser processado e comandos que indicam como ele deve ser processado. Palavras são separadas por um ou mais espaços, parágrafos são separados por uma ou mais linhas em branco. Espaços ou linhas em branco extras não afetam a saída. Toda quebra de linha em um parágrafo (e hifenização, caso esteja habilitada) é feita automaticamente independente de onde ocorre a quebra de linha no editor de texto.

A maioria dos comandos do LaTeX iniciam com o caractere '\'. Uma \ sozinha produz um espaço. Duas \ (\\) indroduzem uma quebra de linha sem iniciar um novo parágrafo. Aguns caracteres são reservados por serem utilizados em comandos são eles:
& Separador de colunas
$ Delimitador de fórmula matematica (na mesma linha do texto)
# Usado para definir o uso de argumentos em certos comandos
% Comentário. O compilador ignora tudo o que vir após o % em uma mesma linha
{ Delimita o ínicio do argumento de um comando
} Delimita o fim do argumento de um comando
_ Subescrito

^ Sobrescrito
~ Espaço inquebrável, usado quando não desejamos que o compilador separe duas palavras (com espaço entre elas) em linhas diferentes

Para exibir esse caracteres no texto eles devem ser escritos com um \ na frente. Portanto digitar no arquivo fonte:

```
\& \$ \# \% \_ \{ \} \^{} \~{}
```

Produzirá a seguinte saída:

& $ # % _ { } ^ ~

A necessidade das chaves nos dois últimos símbolos é porque o uso da \ faz com que eles produzam acentos.

Outro caractere que requer atenção especial são as aspas. Para que produzam o efeito desejado, de "abre" e "fecha" nas aspas duplas elas devem ser abertas com dois acentos graves e fechadas com duas apóstrofes. Para aspas simples o procedimento é análogo.

1.3.1 Definindo e redefinindo comandos

Para criar um novo comando usamos: `\def`{*nome do comando*}{*instruções*}, sendo que o nome do comando deve começar com \ e é como ele será chamado e as instruções são os comandos a serem executados cada vez que ele for chamado. Isto pode ser util para concatenar dois ou mais comandos, ou trocar uma chamada grande de comando por uma menor.

Também é possível redefinir comandos já existentes com:
`\renewcommand`{*cmd*}[*args*][*padrao*]{*definição*}
Onde *cmd* é nome de um comando á existente e que começa com \, *args* é um inteiro de 1 a 9 que indica a quantidade de argumentos recebida pelo comando, se não for especificado é asumida como zero, *padrao* é usado para indicar que o primeiro argumento do comando é opcional e especificar o seu valor e *definição* contém a redefinição dos comandos através de novos comandos, sendo que a ocorrência de um parâmetro da forma `#n` será substituída pelo texto do n-ésimo argumento passado quando o comando for chamado .

1.3.2 Modificadores de texto

É possível modificar a forma, tamanho e tipo das letras do texto através de comandos LaTeX. Apresentamos alguns deles e o resultado produzido:

`\textbf{Texto}` Produz um texto em negrito: **Texto**

`\textit{Texto}` Produz um texto em itálico: *Texto*

`\textrm{Texto}` Produz um texto padrão: Texto

`\texttt{Texto}` Produz um texto estilo máquina de escrever: `Texto`

`\textsf{Texto}` Produz um texto em sanserif: Texto

`\textsc{Texto}` Produz um texto em caixa alta: TEXTO

`\underline{Texto}` Produz um texto sublinhado: Texto

Com exceção deste último, os comandos anteriores funcionam somente no modo texto, para que funcionem no modo matemático, em geral, basta trocar o `text` por `math` nesses comandos. Em relação ao tamanho, da menor para maior:

`\tiny Texto` Produz: Texto

`\scriptsize Texto` Produz: Texto

`\footnotesize Texto` Produz: Texto

`\small Texto` Produz: Texto

`\normalsize Texto` Produz: Texto

`\large Texto` Produz: Texto

`\Large Texto` Produz: Texto

`\LARGE Texto` Produz: Texto

`\huge Texto` Produz: Texto

`\Huge Texto` Produz: Texto

Note que o texto nesses casos não vem entre chaves. Assim que o tamanho da fonte é modificado ela permanece neste tamanho até que seja alterada novamente, no nosso texto após cada exemplo retornamos para `\normalsize`. Tambem é possível utilizar o pacote `color` para produzir um texto colorido. Por exemplo: `\color{blue}Texto`

Produzirá Texto.

Novamente, alterar a cor do texto altera para o restante do documento, portanto foi preciso executar `\color{black}` para voltar à cor original após o exemplo. Outra maneira de fazer isto seria colocando tudo entre chaves, ou seja, `{\color{blue}Texto}` modificará apenas a cor de Texto e não do restante do documento.

Para criar uma nova cor podemos usar o padrão RGB e (preferencialmente) no preâmbulo do documento utilizar:
`\definecolor`{nome da cor}{RGB}{#vermelho,#verde,#azul}".

Por exemplo, o comando `\definecolor{roxinho}{RGB}{140,20,100}` produz esta cor.

Também podemos estar interessados em enfatizar determinada parte do

texto com \emph{*texto*}. Em geral isto implica em deixá-lo itálico, mas em uma linha já itálica ele voltará à formatação original, como podemos ver a seguir:

```
\emph{Enfatizar texto já
enfatizado gera um texto
\emph{normal} como
resultado.}
```

Enfatizar texto já enfatizado gera um texto normal *como resultado.*

1.3.3 Espaçamento

Alguns comandos permitem introduzir ou remover manualmente um espaçamento no texto:

- `\hspace{tamanho}` produz um espaçamento horizontal de tamanho definido pelo usuário. O tamanho pode ser negativo, funcionando como recuo, e deve vir com unidade de medida, por exemplo, cm.
- `\vspace{tamanho}` produz um espaçamento vertical de tamanho definido pelo usuário.
- `\hfill` indroduz espaços entre os textos à direita e a esquerda dele para preencher a largura da página.
- `\vfill` indroduz espaços entre os textos à direita e a esquerda dele para preencher a altura da página.
- `\\[tamanho]` produz um espaçamento vertical de tamanho definido pelo usuário antes de iniciar uma nova linha.
- `\newline` inicia uma nova linha.
- `\newpage` inicia uma nova página.
- `\noindent` remove o espaçamento antes do início do parágrafo

Podemos também usar o pacote `setspace` para modificar o espacamento entre as linhas, o que é feito através dos comandos:

- `\doublespacing`: Linhas com espaçamento duplo.

- `\onehalfspacing`: Linhas com espaçamento 1,5.
- `\singlespacing`: Linhas com espaçamento simples.

Após o uso de um desses comandos espaçamento permanece o especificado, até que um novo comando de espaçamento seja dado.

1.3.4 Links

Podemos fazer referências internas e externas através do pacote `hyperref`. Na sua declaração é possivel passar opções de customização dos links e do documento pdf gerado como:

- **driver**: driver do compilador, pdftex para gerar pdf, este é um item obrigatório.
- **bookmarks** (=true,false): mostra ou esconde os bookmarks quando apresentando o documento.
- **unicode** (=false,true): permite o uso de caracteres unicode nos bookmarks.
- **pdftoolbar** (=true,false): mostra ou esconde a barra de ferramentas do visualizador.
- **pdfmenubar** (=true,false): mostra ou esconde a barra de menu do visualizador.
- **pdffitwindow** (=true,false): ajusta o tamanho inicial da página na janela.
- **pdftitle** (={texto}):Define o título do documento a ser mostrado na seção info do visualizador.
- **pdfauthor** (={texto}): Define o autor do documento a ser mostrado na seção info do visualizador.
- **pdfnewwindow** (=true,false): Indica se uma nova janela deve ser aberta caso um link aponte para fora do documento.

- **colorlinks** (=false,true): Contorna os links com frames coloridos (false) ou colore o texto dos links (true). Essa cor pode ser configurada usando as opções (com os valores default):
  linkcolor (=red): Cor dos links internos (seções, páginas...)
  citecolor (=green): Cor de itens de citação (bibliografia)
  filecolor (=magenta): Cor de links para arquivos color of file links
  urlcolor (=cyan): Cor de links URL

Neste documento declaramos no preâmbulo:

```
\usepackage[pdftex,plainpages=false,pdfpagelabels,pagebackref,
colorlinks=true,citecolor=black,linkcolor=black,urlcolor=black,
filecolor=black,bookmarksopen=true]{hyperref}
```

Para criar um link fazemos `\href{`*url*`}{`*texto*`}`. A url é o endereço do site ou do arquivo que queremos referenciar e o texto é o que será impresso no documento.

1.3.5 Strings Especiais

Existem algumas strings prontas que podem ser chamadas através de comandos. Muitos deles serão apresentados e utilizados ao longo do texto, e neste momento damos destaque para `\LaTeX` que produz a string LaTeX. Por ser um comando ele ignorará os espaços subsequentes, portanto para que seja produzido um espaço em branco após a string (nessa e em todas outras) é preciso incluir uma barra `\` após o comando.

1.4 Ambientes

Um ambiente é uma região do texto que tem um tratamento especial. Um ambiente é iniciado com `\begin` e terminado cum `\end`, seguidos do nome do ambiente entre chaves

1.4.1 Listas

Através do ambiente `itemize` podemos escrever os itens em uma lista com *bullets*:

```
\begin{itemize}
\item Primeiro item
\item Segundo item
\item ...
\end{itemize}
```

- Primeiro item
- Segundo item
- ...

Usar uma lista numerada com `enumerate`:

```
\end{enumerate}
\item Primeiro item
\item Segundo item
\item ...
\end{enumerate}
```

1. Primeiro item
2. Segundo item
3. ...

Ou ainda usar uma lista personalizada com `description`:

```
\begin{description}
\item[1-] Primeiro item
\item[2-] Segundo item
\item[3-] ...
\begin{description}
```

1- Primeiro item

2- Segundo item

3- ...

É possivel ainda aninhar uma lista dentro de outra, começando uma lista nova dentro de um item.

1.4.2 Verbatim

Algumas vezes pode ser de interesse digitar o texto livremente sem que o compilador de fato interprete os comandos, como fizemos várias vezes texto para ilustar exemplos de comandos. Para isto é possível utilizar o ambiente `verbatim`. Todo o texto que estiver dentro deste ambiente é apresentado exatamente como digitado, incluindo quebras de linha e comandos.

Uma maneira mais compacta de produzir o mesmo efeito para textos mais curtos é através do comando `\verb|texto|`. Tudo o que vier entre | é mostrado da forma que foi digitado. O uso do | como delimitador nesse comando não é exclusivo, também funcionaria com outros caracteres desde que não sejam letras, * ou espaço.

1.4.3 Alinhamento

Podemos modificar o alinhamento do texto através de alguns ambientes:

- `flushleft`: Alinha o texto à esquerda.
- `flushright`: Alinha o texto à direita.
- `center`: Centraliza o texto.

1.4.4 Citações

Para dar destaque à citações ou exemplos podemos usar o ambiente `quote`. Um exemplo de saída para o uso deste ambiente pode ser visto a seguir:

> Esta é uma citação.

1.4.5 Modo matemático

O ambiente matemático é bastante utilizado nos documentos acadêmicos de LaTeXda área das ciências exatas pois é nele em que são escritas fórmulas, equações e outros elementos matemáticos. Um pacote bastante utilizado para aumentar ainda mais as possibilidades de escita matemática é o `amsmath`, recomendamos utilizá-lo, muitos dos comandos apresentados aqui assumirão este pacote carregado.

Podemos inserir uma fórmula no meio de um parágrafo com texto através de `$formula$`, ou dar a ela um destaque especial, fazendo com que seja inserida em uma nova linha através de `$$formula$$`. O resultado é o seguinte:

```
Algumas fórmulas simples
como $x^2$ podem vir
no próprio parágrafo,
outras como $$E = mc^2$$
podem merecer um destaque
maior.
```

Algumas fórmulas simples como x^2 podem vir no próprio parágrafo, outras como

$$E = mc^2$$

podem merecer um destaque maior.

Um terceira forma de entrar no modo matemático é utilizando o ambiente `equation` que produz uma equação destacada e numerada (ou sem número se vier acompanhada de um asterisco: `equation*`). Ao invés do número é possível definir um texto específico para a equação através do comando `\tag{`*texto*`}`. Neste caso, quando o a equação for referenciada aparecerá o texto da tag e não um número de equação. Um exemplo:

```
\begin{equation}
A^{i}_{j} = B^{j}_{i}
\tag{transposta}
\end{equation}
```

$$A^i_j = B^j_i \qquad \text{(transposta)}$$

Utilizando a equação no mesmo parágrafo pode ser necessário instruir o compilador para que mostre corretamente a fórmula, colocando índices embaixo do símbolo de somatório e não ao lado, por exemplo. Isto é feito através do comando `\displaystyle` dentro da equação, imediatamente antes do trecho que queremos que seja exibido corretamente.

Mais adiante apresentaremos, sem esgotar o assunto, alguns dos comandos utilizados para produzir escrita matemática.

1.4.6 Tabelas

Uma tabela é especificada pelo ambiente `tabular`. É possivel utilizar linhas horizontais e verticais sem se preocupar com a largura das colunas que é calculada automaticamente pelo LaTeX. A criação de uma tabela é feita da seguinte forma:

`\begin{tabular}`{*espec*}

O argumento *espec* especifica a quantidade e alinhamentos das colunas:

- **|** adiciona uma linha vertical;
- **l** indica uma coluna alinhada à esquerda;
- **r** indica uma coluna alinhada à direita;
- **c** indica uma coluna com texto centralizado;
- **p{largura}** indica uma coluna especial com texto justificadocapaz de quebrar linhas, com a largura especificada com unidade de medida.

Para preencher a tabela usamos `&` para passar para a próxima coluna, `\\` para terminar uma linha e partir para uma nova e `\hline` para criar uma linha horizontal. É possível também adicionar linhas parciais com o comando `\cline{j-i}` onde j e i são as colunas que conterão a linha. Para que a tabela seja criada com sucesso é preciso que todas as linhas contenham o mesmo numero de colunas que o declarado na especificação da mesma.

Para mesclar colunas de células podemos usar o comando
`\multicolumn`{*num*}{*formato*}{*texto*}
que concatena o número de colunas especificado por *num* com o alinhamento especificado por *formato* e possui como conteúdo *texto*

Uma tabela pode ser inserida dentro do ambiente `table` o que faz dela um objeto flutuante. Vantagens de utilizar esse tipo de ambiente é a posição correta da tabela no texto, sem que seja quebrada em das páginas, por exemplo, faz com que a tabela apareça em um índice de tabelas e também a inserção de rótulos e legendas. Para usar este ambiente primeiramente é preciso usar o comando

`\begin{table}`[*pos*]

Onde *pos* indica a posição desejada para se posicionar a tabela vericalmente na página e pode ser:

- **h**, no local onde o texto ocoreu;
- **t**, no topo da página;
- **b**, no fim da página;
- **p**, em uma página especial contendo somente objetos flutuantes;
- **!**, ignora alguns parâmetros internos;

Caso seja passada mais de uma opção, a preferência é da que apareceu primeiro, caso nenhuma opção seja passada é utilizado `[tbp]`. Para adicionar uma legenda usamos ainda dentro do ambiente `table` o comando `\caption[legenda curta]{legenda longa}` onde a legenda longa é a que será exibida junto com a tabela e a legenda curta, opcional, será exibida no índice.

Muitas vezes, conforme o LaTeX não encontra posição adequada para inserir um objeto flutuante, de acordo com o especificado pelo usuário, ele vai acumulando-os em uma pilha e desalocando conforme surgem as oportunidades. Para forçar um esvaziamento do buffer, descarregando os objetos flutuantes acumulados no arquivo, podemos utilizar os comandos `\clearpage` e `\cleardoublepage` que adicionam todas as figuras remanescentes na pilha e iniciam uma nova página.

A seguir apresentamos uma tabela criada como objeto flutuante e os comandos utilizados para que fosse gerada:

Centro	Direita	Esquerda
Números		
1	2	3
4	5	6

Tabela 1.1: Uma tabela simples

```
\begin{table}[ht]
\begin{center}
\begin{tabular}{|c|r|l|}
\hline
Centro & Direita  & Esquerda \\
\hline
\hline
\multicolumn{3}{|c|}{Números}\\
\hline
1 & 2 & 3 \\ \cline{2-3}
4 & 5 & 6 \\
\hline
\end{tabular}
\caption{Uma tabela simples}
\end{center}
\end{table}
```

Para mesclar facilmente as linhas das células podemos usar o pacote `multirow` que possui o comando `\multirow` que funciona de maneira análoga ao `\multicolumn`, sendo que é possível passar como posição um *, indicando que o compilador deve procurar pelo melhor ajuste, e é preciso deixar a posição correspondente da coluna cujas linhas estão sendo mescladas em branco. Um pequeno exemplo:

Centro	Direita	Direita	Esquerda
Números	1	2	3
	4	5	6

```
\begin{tabular}{|c|r|r|l|}
\hline
Centro & Direita  & Direita & Esquerda \\
```

```
\hline
\multirow{2}{*}{Números} & 1 & 2 & 3 \\
& 4 & 5 & 6 \\
\hline
\end{tabular}
```

1.4.7 Figuras

Para inserir figuras no documento é preciso primeiro carregar o pacote `graphicx` com o driver adequado, por exemplo:

`\usepackage[pdftex]{graphicx}`

Com o driver acima podemos incluir figuras em .png e .jpg através do comando:

`\includegraphics[opt]{arquivo}`

Como `opt` podemos passar as opções:

- **width**: Redimenciona a figura para a largura especificada
- **height**: Redimenciona a figura para a altura especificada
- **angle**: Rotaciona a figura no sentido horário (em graus)
- **scale**: Redimensiona a figura na proporção especificada.

E `arquivo` é o nome do arquivo com a figura, com o caminho. Caso não seja especificado o caminho o compilador irá assumir que a figura está no mesmo diretório que o arquivo fonte. É possivel especificar um caminho único para todas as figuras através do comando `\graphicspath{caminho}` no preâmbulo do documento, lembrando que o caminho deve conter a barra no final.

Assim como nas tabelas, existe um ambiente específico para tratar uma figura como um objeto flutuante chamado `figure` que permite inserir legendas, aparecer em um índice de figuras e funciona de maneira simular à vista na seção 1.4.6.

A seguir um exemplo de inserção de figura e os comandos efetuados:

```
\begin{figure}[h!t]
\centering
\includegraphics[scale = 2]{figura.jpg}
\caption{Uma figura qualquer}
\end{figure}
```

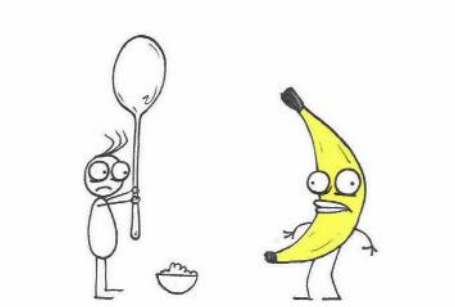

Figura 1.1: Uma figura qualquer

1.4.8 Mini-páginas

Para escrever um pedaço do texto em duas colunas, inclusive podendo inserir figuras e tabelas, ou para dar certos destaque para algum pedaço do texto é possível criar uma mini-página. Cada mini-página é criada da seguinte forma:

`\begin{minipage}[pos]{largura} texto \end{minipage}`

Onde `pos` é o alinhamento vertigal da mini-página (b,t,c) e a `largura` é a largura da página. Para produzir duas colunas podemos fazer por exemplo:

```
\begin{minipage}{5cm}
Primeira coluna
\end{minipage}
\framebox{
\begin{minipage}{6cm}
Segunda coluna
 \end{minipage}
}
```

Produzindo:

Primeira coluna | Segunda coluna

Obs: O comando `\framebox` utilizado na segunda coluna fez com que a página estivesse dentro de uma caixa. Opcionalmente ele pode receber como primeiro argumento a largura da caixa e como segundo a posição.

É possível ainda alternar entre duas e uma coluna com os comandos `\twocolumn[titulo]` e `\onecolumn` onde o título é um argumento opcional para aparecer no início da página. Esses comandos iniciam uma nova página com a quantidade de colunas especificada.

1.4.9 Bibliografia

Para incluir uma seção com as referências bibliográficas devemos usar o ambiente `thebibliography`, seguido de um parâmetro com o número máximo de referências que serão incluídas.

Cada entrada é adicionada precedida do comando:

`\bibitem`[*rótulo*]{*marcador*}

Como `marcador` usamos uma palavra com a qual faremos referencia desta entrada bibliográfica durante o texto através do comando `\ref{marcador}`. Como `rótulo` passamos como deve aparecer a referência no texto, caso não seja especificado são gerados números. Por exemplo:

Temos uma referência em [REFe].

```
Temos uma referência em
\cite{referencia}.
\begin{thebibliography}{99}
\bibitem[REFe]{referencia}
Uma referencia qualquer.
\end{thebibliography}
```

Referências Bibliográficas

[REFe] Uma referencia qualquer.

17

Para trabalhar com muitas referências pode ser indicado utilizar o programa `BibTeX` que vem junto com a distribuição LaTeX. Com ele é possível criar uma base bibliográfica pessoal unificada em um arquivo separado e só incluir no documento as referências que de fato apareceram no texto e também com a formatação correta de título, autor, editora em cada referencia. Existem sites que disponibilizam as entradas no formato do `BibTeX` de diversos

livros, artigos e outros documentos. Para inserir um arquivo de bibliografia no documento fazemos:

```
\bibliographystyle{estilo}
\bibliography{arquivo}
```

Onde `estilo` é o estilo de citação e pode ser *plain*, com referencias em ordem alfabética e rótulos numéricos, *alpha*, com referencias em ordem alfabética e rótulos com nome do autor e ano de publicação, *unsrt*, com referências em ordem de citação e rótulos numéricos, *abbrv*, com referencias em ordem alfabética e rótulos numéricos e compactos (e outros mais vindos de outros pacotes) e arquivo é o nome do arquivo com as entradas, sem extensão.

1.5 Fórmulas matemáticas

Como vimos anteriormente, para digitar fórmulas matemáticas é preciso estar em um ambiente matemático. Veremos agora como escrever o texto dessas fórmulas, assumindo novamente pacotes como `amsmath` e `amssymb` carregados.

Caracteres como + − = < > / : ! ′ | [ ] ( ) podem ser digitados diretamente. Chaves são interpretadas como delimitadores de grupo e para serem impressas devem vir com \ como no modo texto. Espaços em branco são ignorados pelo compilador e, como padrão, todas as letras são escritas em itálico.

1.5.1 Alfabeto grego

A tabela 1.2 apresenta a lista de caracteres do alfabeto grego e o comando correspondente, a primeira coluna são as maiúsculas. As letras que não aparecem na lista são maíusculas correspondentes às maiúsculas do alfabeto romano tradicional.

1.5.2 Símbolos e operadores

Apresentamos agora alguns símbolos bastante frequentes, assumindo que o pacote `amssymb` foi carregado. Existem muitos símbolos definidos e a lista completa com estes símbolos pode ser vista em:

http://www.ctan.org/tex-archive/info/symbols/comprehensive/.

Γ	\Gamma	α	\alpha	κ	\kappa	σ	\sigma
Δ	\Delta	β	\beta	$\varkappa$	\varkappa	ς	\varsigma
Θ	\Theta	γ	\gamma	λ	\lambda	τ	\tau
Λ	\Lambda	δ	\delta	μ	\mu	υ	\upsilon
Ξ	\Xi	ϵ	\epsilon	ν	\nu	ϕ	\phi
Π	\Pi	ε	\varepsilon	ξ	\xi	φ	\varphi
Σ	\Sigma	ζ	\zeta	o	o	χ	\chi
Υ	\Upsilon	η	\eta	π	\pi	ψ	\psi
Φ	\Phi	θ	\theta	ϖ	\varpi	ω	\omega
Ψ	\Psi	ϑ	\vartheta	ρ	\rho	$\digamma$	\digamma
Ω	\Omega	ι	\iota	ϱ	\varrho	∂	\partial

Tabela 1.2: Letras gregas

Um outro site bastante útil é o Detexify[2] que permite ao usuário desenhar o símbolo que deseja conhecer e apresenta algumas sugestões de resultados.

1.5.3 Equações com varias linhas

É possível criar uma equação de multiplas linhas através do ambiente `align` ou invés do `equation`, que funciona de maneira bastante semelhante ao `tabular` e também é um ambiente matemático:

```
\begin{align}
f(x) &= 2x^5 + 3x^4 + x^3
\nonumber \\
&\qquad + 2x^2 + 5x + 8 \\
&= g(x) - h(x)
\end{align}
```

$$f(x) = 2x^5 + 3x^4 + x^3$$
$$+ 2x^2 + 5x + 8 \tag{1.1}$$
$$= g(x) - h(x) \tag{1.2}$$

Pode padrão todas as linhas são numeradas, para suprimir a numeração de uma linha específica basta usar o comando `\nonumber` nesta linha, antes do `\\` como fizemos no exemplo. Assim como em `equation` é possível espeficiar uma tag ao invés de número para a equação.

O pacote `amsmath` fornece ambientes similares para trabalhar com multiplas linhas, são eles: `flalign`, `gather`, `multline` and `split`.

1.5.4 Construção de Fórmulas

Podemos criar expoentes e índices usando os comandos ^ e _, respectivamente. Isso indica que o caractere seguinte ao comando será sobrescrito ou subescrito, para colocar mais de um caractere no índice ou expoente é necessário colocá-los entre chaves. Caso um mesmo objeto possua tanto expoente quanto índice a ordem em que estes elementos aparecem no texto é irrelevante. Por exemplo, utilizando: `\displaystyle\sum_{i = 1}^n` temos $\sum_{i=1}^{n}$.

Frações são criadas com `\frac`{*numerador*}{*denominador*}, ou em sua versão reduzida com `\tfrac` e raízes com `\sqrt[n]{radicando}`. Se o argumento opcional `[n]` for omitido, então a raiz quadrada é gerada. Por exemplo, `$$\frac{\sqrt[3]{xy}}{2}$$` produz:

$$\frac{\sqrt[3]{xy}}{2}$$

Para adicionar limites em somatórios e integrais, em especial quando a fórmula estiver sendo exibida no mesmo parágrafo do texto podemos usar o comando `\limits` para adicionar os limites em cima e abaixo do símbolo, por exemplo, `$\int\limits_a^b f(x)dx$` produz $\int\limits_a^b f(x)dx$.

Podemos ainda estar interessados em empilhar dois símbolos, isto é feito com o comando `\stackrel`. Por exemplo:
`$a \stackrel{\infty}{\rightarrow b$` produz $a \stackrel{\infty}{\rightarrow} b$.

1.5.5 Fórmulas matemáticas

Existem alguns comandos prontos para inserir fórmulas matemáticas. A diferença de utilizá-los ao invés de simplesmente digitar o nome da função no texto é que a função aparecerá em romano normal, enquanto o argumento virá em itálico, enquanto no outro caso seria tudo em itálico, ou seja, o primeiro caso produz: $\sin x$ e o segundo $senx$.

A lista com funções matemáticas é apresentada pela tabela 1.3

E temos ainda algumas outras definições para limites como: `\varinjlim`, `\varprojlim`, `\varliminf`, `\varlimsup` que produzem: $\varinjlim$ $\varprojlim$ $\varliminf$ $\varlimsup$.

Como podemos ver, as funções estão definidas em inglês. Podemos usar o pacote `amsmath` para definir novos operadores através do comando

\arccos	\coth	\hom	\limsup	\sec
\arcsin	\csc	\inf	\ln	\sin
\arctan	\deg	\ker	\log	\sinh
\arg	\det	\lg	\max	\sup
\cos	\dim	\lim	\min	\tan
\cosh	\exp	\liminf	\Pr	\tanh
\cot	\gcd			

Tabela 1.3: Funções Matemáticas

\DeclareMathOperator{comando}{texto},
Por exemplo, declarando no preâmbulo:
\DeclareMathOperator{\sen}{sen}
Produzimos sen x ao digitarmos $\sen x$.

1.5.6 Tamanho automático de delimitadores

Os comandos \left(e \right) são utilizados para ajustar automanticamente os parêntesis para o tamanho da fórmula. Os parentesis podem ser substituídos por outros delimitadores apresentados na tabela 1.4.

() ()	‖‖ \lVert \rVert	⌊⌋ \lfloor \rfloor
[] []	⟨⟩ \langle \rangle	⟮⟯ \lgroup \rgroup
{} \lbrace \rbrace	⌈⌉ \lceil \rceil	⏜ \lmoustache \rmoustache
| \lvert \rvert		
| \vert	\ \backslash	‖ \Arrowvert
‖ \Vert	| \arrowvert	⎪ \bracevert
/ /		

Tabela 1.4: Delimitadores matemáticos

Com excessão das três últimas linhas, todo delimitador iniciado por um \left deve vir com seu \right correspondente. Caso o desejo seja de realmente utilizar somente um dos dois delimitadores, é possivel substituir o outro por um "fantasma", trocando o caractere ou comando do delimitadore por um ponto '.'.

Eventualmente pode ser necessário controlar o tamanho dos delimitadores manualmente. Os comandos \bigl \Bigl \biggl \Biggl \bigr \Bigr

`\biggr` `\Biggr` funcionam para os mesmos delimitadores da tabela 1.4 e produzem na ordem em que apareceram:

$$\biggl(\Biggl(\biggl(\Biggl(x\Biggr)\biggr)\Biggr)\biggr)$$

1.5.7 Matrizes

Podemos escrever facilmente uma matriz através dos ambientes matriciais fornecidos pelo pacote `amsmath`. Os elementos das colunas são separados por `\&` e as linhas por `\\`. Os diferentes comandos são usados para especificar o delimitador a ser utilizado na matriz: `matrix` (nada), `pmatrix` (, `bmatrix` [, `Bmatrix` {, `vmatrix` | and `Vmatrix` ||. Segue um exemplo:

```
$$
\begin{pmatrix}
1 & 2 & 3 \\
4 & 5 & 6 \\
7 & 8 & 9 \\
\end{pmatrix}
$$
```

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

Reticências

Podemos estar interessados em representar reticências dentro (ou fora) de uma matriz. A seguir apresentamos os comandos e os resultados produzidos:

`\ldots` $\ldots$ `\cdots` $\cdots$ `\vdots` $\vdots$ `\ddots` $\ddots$

1.5.8 Espaços

Podemos alterar o espaçamento entre os símbolos dentro do ambiente matemático. Como unidade de medida vamos considerar o `\quad` que equivale ao espaço utilizado pelo caractere `M` na fonte que estiver sendo utilizada. Temos então:

- `\qquad`: dois `\quad`;
- `\,`: 3/18 de um `\quad`;

- `\:`: a 4/18 de um `\quad`;
- `\;`: 5/18 de um `\quad`;
- `\!`: -3/18 de um `\quad`.

1.5.9 Fontes

Podemos, como no modo texto, modificar o estilo de fonte utilizada dentro do ambiente matemático. Seguem alguns exemplos de modificação:
`\mathbf{AaBbCc}` Produz um texto em negrito: **AaBbCc**
`\mathit{AaBbCc}` Produz um texto em itálico: *AaBbCc*
`\mathrm{AaBbCc}` Produz um texto padrão: AaBbCc
`\mathtt{AaBbCc}` Produz um texto estilo máquina de escrever: `AaBbCc`
`\mathsf{AaBbCc}` Produz um texto em sanserif: AaBbCc
`\mathcal{ABC}` Produz um texto em caligrafia: $\mathcal{ABC}$
`\mathbb{ABCR}` Produz um texto em estilo de lousa, requer `amssymb`: $\mathbb{ABCR}$

1.6 Elementos do Documento

Apresentaremos a seguir os principais elementos que compõe um documento.

1.6.1 Título

Podemos inserir informações importantes sobre o documento como título, autor e data logo no início do documento. Para fazer isso devemos usar os comandos:

```
\title{titulo}
\author{autor}
\date{data}
\thanks{agradecimentos}
\maketitle
```

O `titulo` é o título do documento. É possivel inserir quebras de linha e figuras dentro dele. Em `autor` escrevemos o autor do documento, podendo inserir mais de um autor separando-os pelo comando `\and`. A `data`

é opcional, contendo a data em que o documento foi feito. É possível utilizar um comando especial, `\today`, que produz a data do dia atual. Os `agradecimentos` também são opcionais e são gerados como nota de rodapé na página do título. O comando `\maketitle` adiciona os elementos do título em uma página separada, exceto para a classe `article` em que o título é gerado sem criar uma nova página após sua criação. O título que aparece no presente tutorial foi gerado através destes comandos.

1.6.2 Capítulos e seções

Para facilitar a leitura de um documento é recomendável separa-lo em partes. Podemos usar, da divisão mais externa para a mais interna:

```
\part[mini-titulo]{titulo}
\chapter[mini-titulo]{titulo}
\section[mini-titulo]{titulo}
\subsection[mini-titulo]{titulo}
\subsubsection[mini-titulo]{titulo}
\paragraph[mini-titulo]{titulo}
\subparagraph[mini-titulo]{titulo}
```

A numeração das partes, capítulos, seções e subseções é feita automaticamente, conforme esses comandos são inseridos. Para omitir a numeração basta usar * após o comando, por exemplo `\section*{titulo}`. O `titulo` é o título que será apresentado, e `mini-titulo` é uma versão resumida do título, opcional, para ser exibida no índice. A classe `article` não suporta `\chapter`.

Os níveis de cada uma dessas subdivisões são:

`\part`	-1
`\chapter`	0
`\section`	1
`\subsection`	2
`\subsubsection`	3
`\paragraph`	4
`\subparagraph`	5

Essas informações podem ser úteis para modificar a forma de numeração das divisões. O argumento `secnumdepth` do comando `\setcounter` permite alterar a profundidade que vai chegar a numeração, de acordo com os níveis

apresentados na tabela anterior. O padrão é numerar até o nível 2, mas por exemplo, se quisermos apenas partes, capítulos e seções numerados podemos usar `\setcounter{secnumdepth}{1}`. Da mesma forma, para inserir numeração nas subsubseções podemos fazer: `\setcounter{secnumdepth}{3}`

Ao usar o classe `book` temos ainda alguns comandos adicionais, referentes à numeração de capítulos e páginas:

- `\frontmatter`: Deve ser o primeiro comando dado após o inicio do documento. Muda a numeração para numerais romanos e não numera seções e subseções, mas permite que elas apareçam no índice.
- `\mainmatter`:Deve vir antes do primeiro capítulo do livro. Usa uma nova numeração de página com algarismos arábicos.
- `\appendix`:Usado para iniciar a presença de capítulos adicionais, como apendices. Numera os capítulos com letras.
- `\backmatter`:Deve ser inserido antes dos últimos itens do livro, como bibliografia e índice remissivo.

1.6.3 Índices

Alguns simples comandos geram automaticamente os índices de conteúdo, figuras e tabelas. Respectivamente são:

```
\tableofcontents
\listoffigures
\listoftables
```

Os índices são gerados no lugar onde foi dado o comando. Os itens que foram gerados com comandos seguidos de * não entram nos índices, entretanto é possível adiciona-los manualmente através de
`\addcontentsline`{*tipo*}{*profundidade*}{*titulo*}",
imediatamente antes da declaração do item. O *tipo* é o tipo de unidade e pode ser: `toc` para conteúdo de texto, como capítulos e seções, `lof` para figuras e `lot` para tabelas. A *unidade* é a unidade do documento, pode ser `part`, `chapter`, `section`, `subsection`, `subsubsection`, `paragraph`, `subparagraph`, `figure` ou `table`. O *titulo* é o texto que deve ser apresentado na entrada da unidade inserida no índice.

Para que os índices funcionem corretamente é necessario compilar o textos duas vezes, eventualmente três vezes. O compilador indicará esta necessidade através de *warnings*.

Assim como a numeração das partes, é possível alterar a profundidade dos itens de texto exibidos no sumário. Isto pode ser feito com o argumento `tocdepth` do comando `\setcounter`, de maneira similar à vista na seção 1.6.2.

1.6.4 Cabeçalhos e rodapés

A customização dos cabeçalhos e rodapés, na parte direita, esquerda e central, pode ser feita através do uso do pacote `fancyhdr`. Além de inserir textos fixos, ou mesmo figuras, é possível inserir informação variável como número de página ou nome da seção. Alguns exemplos:

- `\leftmark`: Nome do capítulo atual (em maiúsculas);
- `\rightmark`: Nome da seção atual (em maiúsculas);
- `\thepage`: Número da página atual;
- `\chaptername`: Nome "capítulo" na linguagem atual;
- `\thechapter`: Número do capítulo atual;
- `\thesection`: Número da seção atual.

Para usar o pacote devemos declarar o pacote, o estilo da página utilizado, que é o do pacote e também remover o cabeçaho e rodapé atuais através dos comandos:

```
\usepackage{fancyhdr}
\pagestyle{fancy}
\fancyhf{}
```

Para que os comandos `\leftmark` e `\rightmark` produzam o texto sem ser em todas maiúsculas devemos digitar antes de `\pagestyle{fancy}`:

```
\renewcommand{\chaptermark}[1]{\markboth{\MakeUppercase{#1}}{}}
\renewcommand{\sectionmark}[1]{\markright{\MakeUppercase{#1}}{}}
```

Podemos configurar a altura do cabeçalho para ter medida *size* através de `\setlength{\headheight}{`*size*`}`.

Por padrão, os cabeçalhos vem com uma linha horizontal separando-os do resto da página. É possível alterar a espessura da linha para `size` , ou mesmo removê-la (passando tamanho 0), e especificar uma linha para o rodapé através dos comandos:

```
\renewcommand{\headrulewidth}{size}
\renewcommand{\footrulewidth}{size}
```

Para alterar a parte esquerda, central e direita do cabeçalho, esquerda, central e direita do rodapé devemos usar respectivamente:

```
\lhead{Cabeçalho da esquerda}
\chead{Cabeçalho do centro}
\rhead{Cabeçalho da direita}
\lfoot{Rodapé da esquerda}
\cfoot{Rodapé do centro}
\rfoot{Rodapé da direita}
```

As classes `book` e `report` possuem opção de página dupla, neste caso, os comandos anteriores funcionam para as páginas ímpares. É possível diferenciar o cabeçalho e rodapé das páginas pares e ímpares passando um texto opcional para as páginas pares através de um texto entre colchetes antes do texto que vem entre as chaves.

Todos os comandos apresentados para uso de cabeçalho e rodapé devem vir no preâmbulo do documento.

1.6.5 Nota de rodapé

Uma nota de rodapé pode ser inserida após a palavra ou frase a qual se refere através do comando: `\footnote`[*numero*]{*texto*}. O *numero* é opcional, e pode ser usado para forçar um determinado numero ao invés do automático que seria gerado pelo compilador o *texto* é o que vai aparecer na nota no final da página.

Exemplo:

```
Temos uma nota aqui.
\footnote{nota de rodapé}
```

Temos uma nota aqui. [a]

[a] nota de rodapé

1.7 Referências Cruzadas

Podemos usar o comando `\label{`*rotulo*`}` para dar um rótulo para diferentes partes do texto como seções, figuras, tabelas, equações, capítulos... Tudo o que possui um rótulo pode ser referenciado através dos comandos `\ref{`*rotulo*`}` e `\pageref{`*rotulo*`}` que imprimem, respectivamente, a numeração do objeto referenciado e o número da página em que aparece. Por exemplo:

```
Colocando um rotulo aqui
\label{sec:exemplo}.
Podemos ver um exemplo de
referência na seção
\ref{sec:exemplo} da
página \pageref{sec:exemplo}.
```

Colocando um rotulo aqui . Podemos ver um exemplo de referência na seção 1.7 da página 28.

Para que as referências funcionem corretamente é necessário compilar o documento mais de uma vez.

Se, por algum motivo, não se sabe se o rótulo estará definido ou não é possível utilizar o pacote `\hyperref` e o comando `\hyperref[`*rotulo*`]{`*texto*`}`. Caso o *rotulo* esteja definido o documento será criado com um link para o objeto no *texto*, caso contrário imprimirá somente o *texto*, sem link.

Capítulo 2

LaTeX nem tão Básico

2.1 Separando em vários arquivos

Para grandes documentos pode ser mais organizado quebrar o arquivo fonte em vários arquivos separados, em geral, um principal, com o prêmbulo e declarações e um arquivo ou mais para cada capítulo. Os comandos `\include{`*arquivo*`}` e `\input{`*arquivo*`}` adicionam o *arquivo.tex* no momento em que aparecem no texto, com a diferença que o primeiro adiciona em uma nova página e o segundo não.

Como não é possível utilizar mais de uma vez o comando `\begin{document}` em um mesmo documento sem que haja erro de compilação, a simples separação do arquivo em várias não permite que cada arquivo seja compilado individualmente. Caso esse seja o interesse, como compilar cada capítulo de um livro, existem duas opções:

A primeira é utilizar no preâmbulo do documento o comando `\includeonly{\textit{arquivos}}`, onde os arquivos devem vir sem extensão e separados por vírgulas, sem espaços. Apenas os arquivos que estiverem nesta lista e que foram adicionados com `\include` serão processados.

A outra forma é através do pacote `subfiles`. No documento principal declaramos o pacote adicionamos os arquivos com `\subfile{`*arquivo*`}`. Os subdocumentos devem ter a seguinte estrutura:

```
\documentclass[nome do arquivo principal.tex]{subfiles}
\begin{document}
conteúdo...
\end{document}
```

Com isso podemos chamar o LaTeXpara compilar tanto o arquivo principal quanto os subarquivos em documentos separados.

2.2 Macros e condições

O pacote `etoolbox` fornece uma série de ferramentas e macros. Uma bastante útil é o `\ifdef` que executa determinada ação caso a condição seja verdadeira e outra se for falsa. Seu uso é:
`\ifdef`{*comando*}{*verdadeiro*}{*falso*}
O comando pode ser definido no texto através de um `\def`.

2.3 Inserindo código fonte

O pacote `listingsutf8`, que vem junto com o pacote `oberdiek` (essa informação pode ser útil para instalá-lo) permite inserir um arquivo de código diretamente no documento.

Para adicionar o arquivo *fonte.ext* usamos:
`\lstinputlisting`[*opt*]{*fonte.ext*} Algumas das opções são:

- language: Linguagem do fonte, por exemplo:Octave, Python, C, R...
- inputencoding: encoding do arquivo fonte, pode ser passado mais de um, por exemplo utf8/latin1.
- showspaces: variável booleana que indica se os espaços devem ser mostrados como undersores.
- showstringspaces: variável booleana que indica se os espaços entre strings devem ser mostrados como undersores.
- breaklines: variável booleana que indica se deve utilizar quebra de linha automática.
- label: rótulo para identificar o código fonte.
- caption: legenda do código fonte.

Um exemplo de escolha de opções é:
`[language=Python, inputencoding=utf8/latin1, showspaces =false, showstringspaces=false, breaklines=true, label=fonte]`

2.4 Inserindo arquivos em pdf

O pacote `pdfpages` permite incluir páginas ou um documento inteiro em .pdf dentro do documento. Isso deve ser feito através do comando:
`\includepdf`[*pages=i-j*]{*arquivo.pdf*}!

No nome do arquivo a ser inserido neste caso é *arquivo.pdf* e o intervalo de páginas vai de i a j. Para incluir o documento inteiro basta passar somente $-$.

2.5 Algoritmos

O pacote `algorithm2e` serve para colocar algoritmos no documento e permite usar português como idioma.

Apresentamos a seguir um código utilizado para criar um algoritmo genérico e o resultado produzido. Para maiores detalhes sobre o uso deste pacote sugerimos a leitura da documentação do mesmo disponível em

```
\begin{algorithm}[H]
\SetAlgoLined
\Entrada{Conjunto de dados $X$ \\
\Saida{Vetor resposta $Y$}
\Inicio{
   Atribua valores iniciais para as variáveis \\
  \Repita{que alguma certa condição seja verdadeira}{
    \Para{$i \leftarrow 1$ a $N$}{
        ação 1 \\
        ação 2 \\
       }
   }
}
\label{alg:algoritmo}
\caption{Este é um algoritmo}
\end{algorithm}
```

Algoritmo 2.1: Este é um algoritmo

Entrada: Conjunto de dados X
Saída: Vetor resposta Y

```
1 início
2 |  Atribua valores iniciais para as variáveis
3 |  repita
4 |  |  para i ← 1 a N faça
5 |  |  |  ação 1
6 |  |  |  ação 2
7 |  |  fim
8 |  até que alguma certa condição seja verdadeira;
9 fim
```

2.6 Índice Remissivo

Para criar um índice remissivo devemos usar o pacote `makeidx` e dar o comando `\makeindex` no preâmbulo do documento para inicializa-lo.

Cada entrada do índice é adicionada logo após a ocorrência da mesma da seguinte forma:
`\index{`*chave*`}`, onde a *chave* é o texto que aparecerá no índice (com a página correspondente). Se uma mesma chave aparecer mais de uma vez no texto ela aparecerá uma única vez no índice, com os números das paginas das ocorrências. É possível incluir subentradas e subsubentradas de uma entrada colocando '`!`' entre elas:
`\index{`*entrada*`!`*subentrada*`!`*subsubentrada*`}`.

Para imprimir o índice no documento com todas as entradas, geralmente no final do texto, utilizamos `\printindex`.

Para que o índice seja exibido corretamente é preciso interpretar o arquivo de índice .idx criado pelo LaTeX através do programa `makeindex` que vem junto com a distribuição. Ele deve ser chamado com o nome do arquivo de índice, geralmente o mesmo que o fonte do texto, sem extensão como argumento, entre as compilações do texto pelo LaTeX.

2.7 Carregando a hifenização corretamente

Pode ser que o compilador precise "aprender" a hifenizar algumas palavras desconhecidas. Isto pode ser feito atravéz do comando `\hyphenation`, passando a separação correta das palavras com hífens e separando palavras com vírgula, por exemplo:

`\hyphenation`{pa-la-vra1, pa-la-vra2}

Além disso, algumas vezes, mesmo utilizando o pacote `babel` com o idioma em português do brasil o texto pode apresentar alguns problemas de hifenização, como por exemplo o warning:

```
"No hyphenation patterns were loaded for(babel)
the language 'Portuguese'(babel)
I will use the patterns loaded for \language=0 instead"
```

Apesar de gerar o documento, a hifenização não estará correta. Para resolver o problema é preciso habilitar ou instalar o idioma na sua distribuição LaTeX. Este processo é um pouco dependente de distribuição, vamos apresentar brevemente como proceder no TeXLive e no MikTeX:

TeXLive

Em geral o problema é solucionado instalando o seguinte pacote através do terminal:
`texlive-lang-portuguese`.

MikTeX

Junto com o MikTeX (2.8) vem um programa de configurações `mo_admin.exe` que aparece como *settings* dentro da pasta *Maintenance* do MikTeX do menu iniciar.

A figura 2.1 mosta a tela do programa. É preciso certificar que a caixa com `portuguese` está checada.

Após selecionar a linguagem desejada e dar `OK` o programa avisará que é preciso reconstruir os arquivos de formato, basta dar `OK` e a atualização será feita automaticamente.

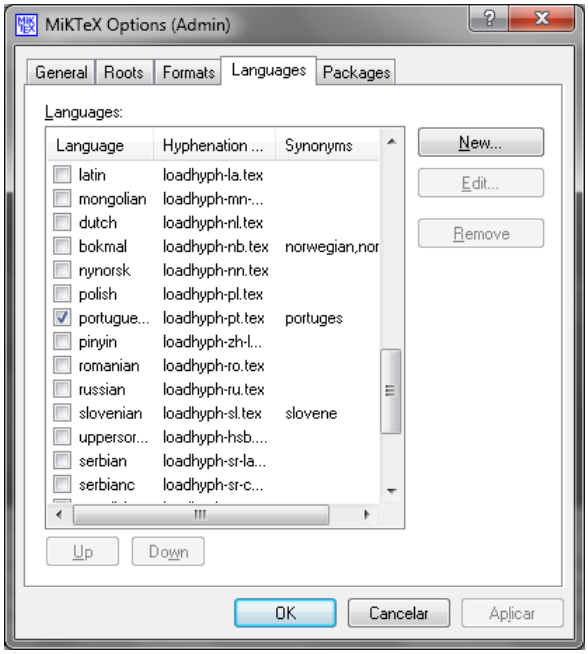

Figura 2.1: Corrigindo a hifenização no windows

Capítulo 3

Miscelanea

3.1 Apresentações

Uma maneira bastante simples e elegante de criar apresentações com o LaTeX é utilizando o pacote `beamer`. Primeiramente, devemos trocar a classe do documento para `beamer`.

Podemos criar as informações do documento através dos comandos `\author{...}`, `\title{...}`, `\institute{...}`, `\subtitle{...}`, `\date{...}`. Essas informações são impressas com o comando `\titlepage` e também são disponibilizadas no rodapé das páginas dos slides, podendo apresentar uma versão abreviada das informações caso sejam passadas informações com os colchetes opcionais.

Beamer vem com uma série de temas que permitem modificar a aparência dos slides

Cada página é criada através do ambiente `frame`. Podemos passar a opção `[plain]` para remover toda a decoração do frame.

3.2 Teoremas

Podemos criar um ambiente para escrita de teoremas, axiomas, etc, através do comando `\newtheorem`{*chave*}[*contador*]{*texto*}[*secao*]
no preâmbulo do documento.

O *chave* é uma chave para identificar o teorema, *contador* é a chave de um teorema já definido, indicando que a numeração deste novo teorema deve seguir a do teorema especificado e é opcional. O *texto* é o texto com o nome

do teorema que será impresso, por exemplo, Teorema, Corolário, Axioma... e, finalmente, *secao* é opcional e refere-se à unidade seccional a qual o teorema deve obter sua numeração (section, chapter, ...).

Após definido o teorema ele pode ser utilizado no texto através de:

```
\begin{chave}[descr]
Este é um teorema
\end{chave}
```

Sendo que o parâmetro opcional `descr` é uma descrição breve, como título, autor ou data do teorema, para aparecer entre parênteses antes do texto do teorema.

O pacote `amsthm` permite ainda configurar o estilo de teorema com as opções: `definition` (título em negrito), `plain` (título em negrito e corpo em itálico) ou `remark` (título em itálico) e também fornece um ambiente `proof` que pode ser utilizado para demonstrações.

Vamos ilustrar o uso desses comandos com alguns exemplos:

Primeiro definimos:

```
\theoremstyle{definition} \newtheorem{teo}{Teorema}
\theoremstyle{plain} \newtheorem{cor}[teo]{Corolário}
\theoremstyle{remark} \newtheorem{nota}{Nota}[section]
```

E então podemos usar nossos teoremas:

```
\begin{teo} \label{teo:1}
Este é um teorema simples.
\end{teo}
\begin{cor}[2012]
Uma consequência imediata
do teorema \ref{teo:1}.
\end{cor}
\begin{nota}
Uma observação simples
\end{nota}
\begin{proof}
Basta ver que
$$2 + 2 = 4$$
\end{proof}
```

Teorema 1. Este é um teorema simples.

Corolário 2 (2012)**.** *Uma consequência imediata do teorema 1.*

Nota 3.2.1. Uma observação simples

Demonstração. Basta ver que

$$2 + 2 = 4$$

□

Podemos ainda utilizar o comando `\qedhere` para exibir o símbolo de fim da demonstração na linha atual e não em uma linha linha nova, que é o padrão do ambiente `proof`.

Para criar teoremas sem numeração podemos usar `\newtheorem*`

3.3 Gerando código em html

Eventualmente pode ser de interesse gerar uma versão em .html do documento. Existem alguns programas para isso e sugerimos aqui o `tex4ht`. Para isso o documento deve ser compilado com o programa `htlatex` ao invés de `pdflatex`.

A instalação do `tex4ht` pode ser um pouco trabalhosa, e também requer alguns programas adicionais como o GhostScript. Um manual de instalação no windows pode ser encontrado aqui.

Este pacote apresenta alguns problemas de compatibilidade com outros pacotes, como por exemplo o `hypcap`, então, para permitir que um mesmo arquivo gere facilmente tanto .pdf quanto .html utilizamos os seguintes comandos no preâmbulo:

```
\makeatletter
\@ifpackageloaded{tex4ht}{
 \DeclareGraphicsExtensions{.png,}
}{
\usepackage{pacote incompativel}
}
\makeatother
```

O comando `\DeclareGraphicsExtensions` permite que as figuras sejam inseridas sem precisar incluir a extensão, no caso, png.

O comando define `\makeatletter` temporariamente o caractere `@` como um caractere, para que possa ser utilizado para alterar em comandos e macros internas, e depois `\makeatother` faz com que ele volte a ter sua função original.

Finalmente, a macro `\@ifpackageloaded`{pacote}{verdadeiro}{falso}. Executa determinada ação se o pacote foi carregado e outra se não foi. No nosso caso, a condição será verdadeira caso o documento esteja sendo gerado com o `htlatex`.

3.4 Unidades de Medida

Muitos comandos recebem alguma medida como parâmetro. Além das unidade tradicionais como milímetros (`mm`) e centímetros (`cm`) podemos passar medidas referentes ao texto ou à página, por exemplo:

- `\linewidth`: Representa o tamanho atual da linha, considerando o ambiente em que está inserida, como uma lista, e é variável.
- `\textwidth`:É a largura total do bloco de texto e é constante.
- `\columnwidth`: É a largura total da coluna, é diferente da largura do texto para mais de uma coluna, e é constante.

Esses comandos podem ser utilizados como se fossem unidades de medida, `0.5\linewidth`, significa metade da linha, por exemplo.

Referências Bibliográficas

[1] AZEREDO, A. D., Pequeno Manual de Introdução ao LaTeX. (2001) *Disponível em: http://paginas.ucpel.tche.br/∼lebrao/manual.pdf*

[2] OETIKER, T., PARTL, H., HYNA, I., SCHLEGL, E. (2008) The Not So Short Introduction to LaTeX. *Disponível em: http://tobi.oetiker.ch/lshort/lshort.pdf*

[3] SANTOS, R. J. Introdução ao LaTeX. (2002) *Disponível em: http://www.mat.ufmg.br/∼regi/topicos/intlat.html*

[4] VAZ, C. L. D. Aprendendo LaTeX. (2001) *Disponível em: http://www.lac.inpe.br/∼margarete/...0/apostila_latexpdf.pdf*

[5] LaTeX Wikibook *Disponível em: http://en.wikibooks.org/wiki/LaTeX*