

Yihui Xie

***bookdown: livros de autoria e
Documentos Técnicos com R***
Remarcação

A Shao Yong (ÿÿ), por
compartilhar uma alegria secreta com palavras simples;

Quando a lua chega ao céu, quando o vento vem acima da
água. Geralmente significado claro, poucas pessoas sabem disso.

e

A Hongzhi Zhengjue (ÿÿÿÿ), por
compartilhar a paz de uma vida que termina com palavras simples.

Céu de sonho, sessenta e sete anos;
pássaros brancos submersos, águas de outono.

Conteúdo

Lista de Figuras	vii
Lista de mesas	ix
Prefácio	XI
Sobre o autor	xix
1 Introdução	1
1.1 Motivação	2
1.2 Comece.	3
1.3 Uso.	4
1.4 Duas abordagens de renderização	8
1.5 Algumas dicas	9
2 Componentes	11
2.1 Sintaxe de Markdown	11
2.1.1 Formatação em linha	11
2.1.2 Elementos de nível de bloco	12
2.1.3 Expressões matemáticas	14
2.2 Extensões de Markdown por bookdown	15
2.2.1 Número e equações de referência	15
2.2.2 Teoremas e demonstrações	18
2.2.3 Cabeçalhos especiais.	25
2.2.4 Referências de texto	26
2.3 Código R	27
2.4 Figuras	28
2.5 Tabelas.	33
2.6 Referências cruzadas	38
2.7 Blocos personalizados	39

2.8 Citações.	39
2.9 Índice.	43
2.10 Widgets HTML.	44
2.11 Páginas da Web e aplicativos Shiny	47
3 Formatos de Saída	49
3.1 HTML.	50
3.1.1 Estilo GitBook	50
3.1.2 Estilo Bootstrap de três colunas	59
3.1.3 O estilo Bootstrap padrão.	68
3.1.4 Estilo tufado.	72
3.2 LaTeX/PDF	73
3.3 Livros Eletrônicos	75
3.3.1 EPUB.	75
3.3.2 MOBI.	76
3.4 Um único documento.	76
4 Personalização	79
4.1 Opções YAML	79
4.2 Tematização	82
4.3 Modelos.	84
4.4 Configuração.	86
4.5 Internacionalização	87
5 Edição	91
5.1 Construir o livro	91
5.2 Visualizar um capítulo.	93
5.3 Sirva o livro.	93
5.4 IDE do RStudio.	95
5.5 Colaboração.	98
6 Publicação	101
6.1 RStudio Connect	101
6.2 Netlify Drop	102
6.2.1 A sequência de pipeline de construção e implantação	102
6.2.2 Antes de começar	103
6.2.3 Construa seu livro.	103
6.2.4 Implante seu site.	103

Conteúdo

6.2.5 Opcional: atualize seu site	104
6.2.6 Opcional: altere o subdomínio padrão	104
6.2.7 Desvantagens e alternativas	105
6.3 GitHub	105
6.4 Recursos para publicação em HTML	110
6.4.1 Páginas HTML 404	110
6.4.2 Metadados para compartilhamento	112
6.5 Editores	112
Apêndice	117
A Ferramentas de	117
Software	
A.1 Pacotes R e R :	117
A.2 Pandoc	118
A.3 LaTeX	119
B Uso do software	121
B.1 knitr	121
B.2 R Markdown	123
Perguntas frequentes	127
Bibliografia	129
Índice	131

Lista de Figuras

2.1 Um exemplo de figura com a proporção especificada, largura, e alinhamento.	30
2.2 Exemplo de figura com largura relativa de 70%.	31
2.3 Duas parcelas colocadas lado a lado.	31
2.4 Três logotipos knitr incluídos no documento de um arquivo de imagem PNG externo.	32
2.5 Um widget de tabela renderizado através do pacote DT.	45
2.6 Um aplicativo Shiny criado através da miniUI pacote; você pode ver uma versão ao vivo em https://yihui.shinyapps.io/miniUI/	48
3.1 A barra de ferramentas do GitBook.	55
3.2 Página inicial de um livro com o Bootstrap de três colunas estilo.	60
3.3 Captura de tela do RStudio Project Wizard para criar um novo projeto de bookdown.	62
3.4 Um bloco de texto explicativo especial.	65
5.1 O complemento RStudio para ajudar a inserir matemática LaTeX.	97
5.2 O complemento RStudio para ajudar a inserir citações.	98
5.3 Uma página de livro com uma área de discussão.	100
6.1 Captura de tela da caixa de atualização de implantação de arrastar e soltar Netlify.	104
6.2 Captura de tela de uma página 404 de exemplo.	111
6.3 Capturas de tela mostrando a imagem de capa, título e descrição de um livro HTML quando o link é compartilhado no Facebook e LinkedIn (esquerda) e no Twitter (direita). . .	112

Lista de mesas

2.1 Ambientes de teoremas em bookdown	19
2.2 Uma tabela das primeiras 10 linhas dos dados mtcars.	34
2.3 Um Conto de Duas Mesas.	34
2.4 Uma tabela gerada pelo pacote longtable.	35

Prefácio

Este pequeno livro apresenta um pacote R, **bookdown**, para alterar seu fluxo de trabalho de escrever livros. Deve ser tecnicamente fácil escrever um livro, visualmente agradável de ver o livro, divertido de interagir com o livro, conveniente para navegar pelo livro, simples para os leitores contribuir ou deixar comentários para o(s) autor(es) do livro e mais importante É importante ressaltar que os autores nem sempre devem se distrair com detalhes de composição.

O pacote **bookdown** é construído em cima do R Markdown (<http://rmarkdown.rstudio.com>), e herda a simplicidade da sintaxe Markdown (você pode aprender o básico em cinco minutos; consulte a Seção 2.1), bem como a possibilidade de vários tipos de formatos de saída (PDF/HTML/Word/...). Ele também adicionou recursos como saída HTML de várias páginas, numeração e referências cruzadas de figuras/tabelas/seções/equações, inserção de partes/apêndices e importou o estilo GitBook (<https://www.gitbook.com>) para criar páginas de livro HTML elegantes e atraentes. Este livro em si é um exemplo de como você pode produzir um livro a partir de uma série de documentos R Markdown, e tanto a versão impressa quanto a versão online podem parecer profissionais.

Você pode encontrar mais exemplos em <https://bookdown.org>.

Apesar do nome do pacote conter a palavra “book”, **bookdown** não é apenas para livros. manual de software, uma tese ou mesmo um diário. Na verdade, muitos recursos de **bookdown** também se aplicam a documentos R Markdown únicos (consulte a Seção 3.4).



A versão online deste livro está licenciada sob a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International Li

censo¹. Você pode comprar uma cópia impressa da Chapman & Hall² ou Ama Sol.

Por que ler este livro

Podemos escrever um livro em um formato de origem e gerar a saída para vários formatos? Tradicionalmente, os livros geralmente são escritos com LaTeX ou Microsoft Word. Qualquer uma dessas ferramentas fará com que escrever livros seja uma viagem só de ida e você não pode voltar atrás: se você escolher o LaTeX, normalmente você acabar apenas com um documento PDF; se você trabalha com o Word, você está provavelmente terá que ficar no Word para sempre, e também pode perder os muitos recursos úteis e a bela saída em PDF do LaTeX.

Podemos nos concentrar em escrever o conteúdo sem nos preocuparmos muito com tipografia? Parece uma contradição natural entre conteúdo e aparência, e sempre temos que equilibrar nosso tempo gasto nesses dois aspectos. Ninguém pode comer um bolo e comê-lo também, mas isso não significa que não pode ter metade e comer metade. Queremos que nosso livro pareça razoavelmente bonito, e também queremos focar no conteúdo. Uma possibilidade é desista do PDF temporariamente, e o que você pode ter em troca é uma visualização do seu livro como páginas da web em HTML. O LaTeX é uma excelente ferramenta de configuração de tipos, mas você pode ser facilmente enterrado nos inúmeros comandos e detalhes de composição do LaTeX enquanto estiver trabalhando no livro. Isso é tão difícil evitar pré-visualizar o livro em PDF, e infelizmente também tão comum encontrar certas palavras que ultrapassam a margem da página, certas figuras flutuam para uma página aleatória, cinco ou seis palavras perdidas no final de um capítulo orgulhosamente ocupa uma página totalmente nova, e assim por diante. Se o livro está para ser impresso, teremos que lidar com essas questões eventualmente, mas não vale a pena se distrair repetidamente enquanto você está escrevendo o conteúdo do livro. O fato de que a sintaxe Markdown é mais simples e tem menos recursos que o LaTeX também ajuda você a se concentrar em o conteúdo. Você realmente precisa definir um novo comando como `\myprecious{}` que aplique `\textbf{\textit{\textsf{}}}` ao seu texto? Será que o

¹<http://creativecommons.org/licenses/by-nc-sa/4.0/>

²<https://www.crcpress.com/product/isbn/9781138700109>

a letra “R” deve ser incluída em `\proglang{}` quando os leitores podem facilmente descobrir que ela representa a linguagem R? Não faz muita diferença se tudo, ou nada, precisa da atenção do leitor.

Os leitores podem interagir com exemplos em nosso livro enquanto o lêem? A resposta é certamente não se o livro for impresso em papel, mas é possível se o seu livro tiver uma versão HTML que contenha exemplos ao vivo, como aplicativos Shiny (<https://shiny.rstudio.com>) ou HTML wid gets (<https://htmlwidgets.org>). Por exemplo, os leitores podem imediatamente saber o que acontece se eles alterarem certos parâmetros de uma estatística modelo.

Podemos obter feedback e até mesmo contribuições dos leitores à medida que desenvolvemos o livro? Tradicionalmente, o editor encontrará um pequeno número de revisores anônimos para revisar seu livro. Os revisores geralmente são úteis, mas você ainda pode perder a sabedoria de leitores mais representativos.

É tarde demais depois que a primeira edição é impressa, e os leitores podem precisar esperar alguns anos antes que a segunda edição esteja pronta. Há algumas plataformas da web que facilitam para as pessoas fornecerem feedback e contribuir com seus projetos. GitHub (<https://github.com>) é um exemplo proeminente. Se alguém encontrar um erro de digitação em seu livro, ele pode simplesmente corrija-o on-line e envie a alteração de volta para sua aprovação.

É uma questão de clicar em um botão para mesclar a alteração, sem fazer perguntas ou enviar e-mails para frente e para trás. Para poder usar essas plataformas, você precisa aprender o básico de ferramentas de controle de versão como GIT, e seu os arquivos de origem do livro devem estar em texto simples.

A combinação de R (<https://www.r-project.org>), Marcação, e Pandoc (<http://pandoc.org>) torna possível ir de um formato de origem simples (R Markdown) para vários formatos de saída possíveis (PDF, HTML, EPUB e Word, etc.). O pacote **bookdown** é baseado no R Markdown e fornece formatos de saída para livros e artigos de formato longo, incluindo o formato GitBook, que é um formato de saída HTML com uma interface de usuário útil e bonita. Isso é muito mais fácil de escrever HTML do que LaTeX, então você sempre pode visualizar seu livro em HTML e trabalhe em PDF depois que o conteúdo estiver quase pronto. Exemplos ao vivo podem ser facilmente incorporados em HTML, o que pode tornar o livro mais atraente e útil. R Markdown é um formato de texto simples, para que você também possa aproveitar os benefícios do controle de versão, como colaboração

orando no GitHub. Também tentamos portar alguns recursos importantes do LaTeX para HTML e outros formatos de saída, como numeração de figuras/tabelas e referências cruzadas.

Em suma, você apenas prepara alguns capítulos de livros do R Markdown, e o livro pode ajudá-lo a transformá-los em um belo livro.

Estrutura do livro

Os capítulos 1 e 2 apresentam o uso e a sintaxe básicos, que devem ser suficientes para que a maioria dos leitores comece a escrever um livro. Os capítulos 3 e 4 são para aqueles que desejam ajustar a aparência de seus livros. Eles podem parecer muito técnicos se você não estiver familiarizado com HTML/CSS e LaTeX. Você não precisa ler estes dois capítulos com muita atenção pela primeira vez. Você pode aprender o que pode ser alterado e voltar mais tarde para saber como. Para o Capítulo 5, os detalhes técnicos não são importantes, a menos que você não use o RStudio IDE (Seção 5.4). Da mesma forma, você pode se sentir sobrecarregado com os comandos apresentados no Capítulo 6 para publicar seu livro, mas, novamente, tentamos facilitar a publicação de seu livro online por meio do IDE do RStudio. Os comandos e funções personalizadas são apenas para quem optar por não usar o serviço do RStudio ou quiser entender os detalhes técnicos.

Para resumir, este livro é uma referência abrangente do pacote **bookdown**. Você pode seguir a regra 80/203 ao lê-lo. Algumas seções estão lá por uma questão de completude, e nem todas as seções são igualmente úteis para o(s) livro(s) específico(s) que você pretende escrever.

Informações e convenções de software

Este livro é principalmente sobre o pacote R **bookdown**, então você precisa pelo menos instalar o R e o pacote **bookdown**. No entanto, seu livro

³ https://en.wikipedia.org/wiki/Pareto_principle

não precisa estar relacionado à linguagem R. Ele pode usar outras linguagens de computação (C++, SQL, Python e assim por diante; veja o Apêndice B), e pode até ser totalmente irrelevante para a computação (por exemplo, você pode escrever um romance ou uma coleção de poemas). As ferramentas de software necessárias para construir um livro são apresentadas no Apêndice A.

As informações da sessão R ao compilar este livro são mostradas abaixo:

```
SessãoInfo()
```

```
## R versão 4.2.1 (23/06/2022)
## Plataforma: x86_64-apple-darwin17.0 (64 bits)
## Rodando em: macOS Big Sur ... 10.16
##
## Produtos de matriz: padrão
##
## local:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF
8
##
## pacotes básicos anexados: ## [1]
estatísticas           gráficos grDevices utils                  conjuntos de dados
## [6] métodos básicos
##
## carregado por meio de um namespace (e não anexado):
## [1] tricô_1.39          ferramentas_4.2.1      miniUI_0.1.1.1
## [4] htmltools_0.5.3     bookdown_0.27.3      shiny_1.7.2
## [7] rmarkdown_2.14
```

Não adicionamos prompts (> e +) ao código-fonte R neste livro e comentamos a saída de texto com dois hashes ## por padrão, como você pode ver nas informações da sessão R acima. Isso é para sua conveniência quando você deseja copiar e executar o código (a saída de texto será ignorada, pois está comentada). Os nomes dos pacotes estão em negrito (por exemplo, **rmarkdown**), e o código embutido e os nomes dos arquivos são formatados em uma fonte de máquina de escrever (por exemplo, knitr::knit('foo.Rmd')). Os nomes das funções são seguidos por parênteses (por exemplo, bookdown::render_book()). O operador de dois pontos :: significa acessar um objeto de um pacote.

Agradecimentos Em

primeiro lugar, gostaria de agradecer ao meu empregador, RStudio, por me dar a oportunidade de trabalhar neste projeto emocionante. Eu esperava trabalhar nele quando vi o projeto GitBook pela primeira vez em 2013, porque imediatamente percebi que era um belo estilo de livro e havia muito mais poder que poderíamos adicionar a ele, a julgar pela minha experiência de escrever o livro de [tricô](#) (Xie, 2015) e lendo outros livros. R Markdown tornou-se maduro depois de dois anos e, felizmente, **bookdown** se tornou meu trabalho oficial no final de 2015. Não há muitas coisas no mundo melhores do que o fato de seu trabalho ser seu hobby (ou vice-versa). Gostei muito de brincar com bibliotecas JavaScript, pacotes LaTeX e infinitas expressões regulares em R. Honestamente, também devo agradecer ao Stack Overflow (<https://stackoverflow.com>), e acredito que todos vocês sabem o que quero dizer,⁴ se você já escreveu algum código de programa.

Este projeto certamente não é o esforço de uma única pessoa. Vários colegas do RStudio me ajudaram ao longo do caminho. Hadley Wickham forneceu uma enorme quantidade de feedback durante o desenvolvimento do **bookdown**, enquanto trabalhava em seu livro *R for Data Science* com Garrett Grolemund. JJ Allaire e Jonathan McPherson forneceram muita ajuda técnica diretamente para este pacote, bem como suporte no IDE do RStudio. Jeff Allen, Chaita Chaudhari e a equipe do RStudio Connect têm mantido o <https://bookdown.org> local na rede Internet. Robby Shaver desenhou uma bela imagem de capa para este livro. Tanto Hadley Wickham quanto Mine Cetinkaya Rundel revisaram o manuscrito e me deram muitos comentários úteis. Tareef Kawaf deu o seu melhor para me ajudar a me tornar um engenheiro de software profissional. É uma benção trabalhar nesta empresa com pessoas entusiasmadas e inteligentes. Lembro-me de uma vez que disse a Jonathan: “ei, encontrei um problema no cache de dependências de widgets HTML e finalmente descobri uma possível solução”. Jonathan pegou sua cerveja e disse: “Já resolvi”. “Ah, legal, legal.”

Também recebi muitos comentários de autores de livros fora do RStudio, incluindo Jan de Leeuw, Jenny Bryan, Dean Attali, Rafael Irizarry,

⁴ <http://bit.ly/2cWbiAp>

Michael Love, Roger Peng, Andrew Clark e assim por diante. Alguns usuários também contribuíram com código para o projeto e ajudaram a revisar o livro. Aqui está uma lista de todos os contribuidores: <https://github.com/rstudio/bookdown/graphs/contributors>. É bom quando você inventa uma ferramenta e percebe que também é o beneficiário de sua própria ferramenta. Como alguém que ama o modelo de pull request do GitHub, eu gostaria que os leitores não precisassem me enviar um e-mail se houvesse um erro de digitação ou um erro óbvio no meu livro, mas que pudessem corrigi-lo por meio de um pull request. Isso foi possível no **bookdown**. Você pode ver quantos pull requests em erros de digitação eu juntei : <https://github.com/rstudio/bookdown/pulls> . É bom ter tantos verificadores ortográficos humanos cuidadosos terceirizados. Não é que eu não saiba como usar um corretor ortográfico real, mas eu não quero fazer isso antes que o livro esteja terminado, e o malvado Yihui também quer deixar algumas tarefas simples para os leitores para envolvê-los em melhorar o livro.

Callum Webb gentilmente projetou um belo adesivo hexbin para **bookdown**.

O pacote **bookdown** não é possível sem alguns pacotes de software de código aberto. Em particular, Pandoc, GitBook, jQuery e os pacotes R dependentes, sem mencionar o próprio R. Agradeço aos desenvolvedores desses pacotes.

Mudei-me para Omaha, Nebraska, em 2015, e passei um ano no Steeplechase Apartments, onde vivi confortavelmente enquanto desenvolvia o pacote **bookdown** , graças à equipe extremamente simpática e prestativa. Então conheci um corretor de imóveis profissional e inteligente, Kevin Schaben, que encontrei um lar fabuloso para nós em um período de tempo incrivelmente curto, e terminei este livro em nosso novo lar.

John Kimmel, editor da Chapman & Hall/CRC, me ajudou a publicar meu primeiro livro. É um prazer trabalhar com ele novamente. Ele generosamente concordou em me deixar ficar com a versão online deste livro gratuitamente, para que eu possa continuar atualizando-o depois que ele for impresso e publicado (ou seja, você não precisa esperar anos pela segunda edição para corrigir erros e introduzir Novas características). Eu gostaria de poder ter a mente tão aberta quanto ele é quando tenho a idade dele. Rebecca Condit e Suzanne Lassandro revisaram o manuscrito e suas sugestões foram profissionais e úteis. Shashi Kumar resolveu alguns dos meus problemas técnicos com a classe LaTeX da editora (krantz.cls) quando eu estava tentando integrá-la ao **livro-**

baixa. Agradeço também os comentários muito úteis dos revisores Jan de Leeuw, Karl Broman, Brooke Anderson, Michael Grayling, Daniel Kaplan e Max Kuhn.

Por fim, quero agradecer à minha família, em particular, à minha esposa e filho, pelo apoio. O menino de um ano descobriu que meu monitor acende quando ele toca meu teclado, então, ocasionalmente, ele simplesmente entra no meu escritório e pressiona aleatoriamente o teclado quando estou fora.
Não tenho certeza se isso conta como sua contribuição para o livro... @)%&@*

Yihui Xie

Elkhorn, Nebrasca

Sobre o autor

Yihui Xie (<http://yihui.org>) é engenheiro de software no RStudio (<http://www.rstudio.com>). Ele obteve seu PhD do Departamento de Estatística da Iowa State University. Ele está interessado em estatística interativa gráficos e computação estatística. Como um usuário ativo do R, ele autorizou vários pacotes do R, como **knitr**, **bookdown**, **blogdown**, **animation**, **DT**, **tinytex**, **tufte**, **formatR**, **fun**, **mime**, **highr**, **servr** e **Rd2roxygen**, entre os quais o pacote de **animação** ganhou o prêmio John 2009 Prêmio de Software Estatístico M. Chambers (ASA). Ele também é co-autor de um alguns outros pacotes R, incluindo **shiny**, **rmarkdown** e **folheto**.

Em 2006, fundou a Capital da Estatística (<https://cosx.org>), que tornou-se uma grande comunidade online de estatísticas na China. Ele iniciou a conferência R chinesa em 2008 e desde então está envolvido na organização de conferências R na China. Durante a sua formação de doutoramento na Iowa State University, ganhou o Prêmio de Computação Estatística Vince Sposito (2011) e o Prêmio Snedecor (2012) no Departamento de Estatísticas.

Ele ocasionalmente reclama no Twitter (<https://twitter.com/xieyihui>), e na maioria das vezes você pode encontrá-lo no GitHub (<https://github.com/yihui>).

Ele gosta de comida picante tanto quanto literatura clássica chinesa.

1

Introdução

Este livro é um guia para autoria de livros e documentos técnicos com R Markdown ([Allaire et al., 2022b](#)) e o pacote R **bookdown** ([Xie, 2022a](#)). Ele se concentra nos recursos específicos para escrever livros, artigos longos ou relatórios, como:

- como compor equações, teoremas, figuras e tabelas e fazer referência cruzada a eles;
- como gerar vários formatos de saída, como HTML, PDF e e-books para um único livro;
- como personalizar os modelos de livro e estilizar diferentes elementos em um livro;
- suporte ao editor (em particular, o RStudio IDE); e
- como publicar um livro.

Não é uma introdução abrangente ao R Markdown ou ao pacote **knitr** ([Xie, 2022b](#)), sobre o qual o **bookdown** foi construído. Para saber mais sobre o R Markdown, consulte a documentação on-line <http://rmarkdown.rstudio.com>. Para **knitr**, veja [Xie \(2015\)](#).

Você não precisa ser um especialista na linguagem R ([R Core Team, 2022](#)) para ler este livro, mas espera-se que você tenha algum conhecimento básico sobre R Markdown e **knitr**. Para iniciantes, você pode começar com as dicas [em https://www.rstudio.com/resources/cheatsheets/.The](#) o apêndice deste livro contém breves introduções desses pacotes de software. Para poder personalizar os modelos e temas do livro, você deve estar familiarizado com LaTeX, HTML e CSS.

1.1 Motivação

Markdown é uma linguagem maravilhosa para escrever documentos relativamente simples que contêm elementos como seções, parágrafos, listas, links e imagens, etc. Pandoc (<http://pandoc.org>) ampliou bastante a sintaxe original do Markdown,¹ e adicionou alguns novos recursos úteis, como notas de rodapé, citações e tabelas. Mais importante, Pandoc torna possível gerar documentos de saída de uma grande variedade de formatos do Markdown, incluindo HTML, LaTeX/PDF, Word e slides.

Ainda faltam alguns recursos úteis no Markdown do Pandoc em o momento em que são necessários para escrever um documento relativamente complicado como um livro, como numeração automática de figuras e tabelas na saída HTML, referências cruzadas de figuras e tabelas e muita controle da aparência das figuras (por exemplo, atualmente é impossível especificar o alinhamento de imagens usando a sintaxe Markdown). Esses são alguns dos problemas que abordamos no **bookdown** pacote.

Sob a restrição de que queremos produzir o livro em vários formatos de saída, é quase impossível cobrir todos os recursos possíveis específicos para esses diversos formatos de saída. Por exemplo, pode ser difícil reinventar um certo ambiente LaTeX complicado na saída HTML usando a sintaxe (R) Markdown. Nossa principal objetivo não é substituir *tudo* pelo Markdown, mas cobrir as funcionalidades *mais* comuns necessárias para escrever um documento relativamente complicado e tornar a sintaxe dessas funcionalidades consistente em todos os formatos de saída, para que você só precisa aprender uma coisa e funciona para todos os formatos de saída.

Outro objetivo deste projeto é facilitar a produção de livros que parecer visualmente agradável. Alguns bons exemplos existentes incluem GitBook (<https://www.gitbook.com>), Tufte CSS (<http://edwardtufte.github.io/tufte-css/>), e Tufte-LaTeX (<https://tufte-latex.github.io/tufte-latex/>). Esperamos integrar esses temas e estilos no **bookdown**,

¹<http://daringfireball.net/projects/markdown/>

1.2 Comece

para que os autores não precisem se aprofundar nos detalhes de como usar uma determinada classe LaTeX ou como configurar CSS para saída HTML.

1.2 Comece

A maneira mais fácil para iniciantes começarem a escrever um livro com R Markdown e **bookdown** é através da demo bookdown-demo em GitHub:

1. Baixe o repositório [GitHub https://github.com/rstudio/bookdown-demo](https://github.com/rstudio/bookdown-demo) como um arquivo Zip,² e descompacte-o localmente.
2. Instale o IDE do RStudio. Observe que você precisa de uma versão superior a 1.0.0. Faça o download da versão mais recente³ se a versão do RStudio for inferior a 1.0.0.
3. Instale o **bookdown** do pacote R :

```
# versão estável no CRAN
install.packages("bookdown")
# ou versão de desenvolvimento no GitHub #
remotes::install_github('rstudio/bookdown')
```

4. Abra o repositório bookdown-demo que você baixou no RStudio clicando em bookdown-demo.Rproj.
5. Abra o arquivo R Markdown index.Rmd e clique no botão Build Book na aba Build do RStudio.



Se você planeja imprimir seu livro em PDF, precisará de uma distribuição LaTeX. Recomendamos que você instale o TinyTeX (que inclui o XeLaTeX): <https://yihui.org/tinytex/>.

²<https://github.com/rstudio/bookdown-demo/archive/main.zip> ³<https://www.rstudio.com/products/rstudio/download/>

Agora você deve ver a página de índice deste livro demo no RStudio Viewer. Você pode adicionar ou alterar os arquivos R Markdown e pressionar o botão Knit novamente para visualizar o livro. Se preferir não usar o RStudio, você também pode compilar o livro pela linha de comando. Consulte a próxima seção para obter detalhes.

Embora você veja alguns arquivos no exemplo de demonstração do livro , a maioria deles não é essencial para um livro. Se você se sentir sobrecarregado com o número de arquivos, pode usar este exemplo mínimo, que é essencialmente um arquivo index.Rmd: <https://github.com/yihui/bookdown minimal>. O exemplo bookdown-demo contém algumas configurações avançadas que você pode querer aprender mais tarde, como personalizar o preâmbulo do LaTeX, ajustar o CSS e construir o livro no GitHub, etc.

1.3 Uso

Um livro **bookdown** típico contém vários capítulos, e um capítulo reside em um arquivo R Markdown, com a extensão de nome de arquivo .Rmd. Cada arquivo R Markdown deve começar imediatamente com o título do capítulo usando o título de primeiro nível, por exemplo, # Título do capítulo. Todos os arquivos R Markdown devem ser codificados em UTF-8, especialmente quando contiverem caracteres de vários bytes, como chinês, japonês e coreano. Aqui está um exemplo (os marcadores são os nomes dos arquivos, seguidos pelo conteúdo do arquivo):

- index.Rmd

```
# Prefácio {-}
```

Neste livro, apresentaremos uma interessante
método.

- 01-intro.Rmd

```
# Introdução
```

1.3 Uso

Este capítulo é uma visão geral dos métodos que propomos para resolver um **“problema importante”**.

- 02-literatura.Rmd

```
# Literatura
```

Aqui está uma revisão dos métodos existentes.

- 03-método.Rmd

```
# Métodos
```

Descrevemos nossos métodos neste capítulo.

- 04-aplicativo.Rmd

```
# Formulários
```

Algumas aplicações significativas são demonstradas neste capítulo.

```
## Exemplo um
```

```
## Exemplo dois
```

- 05-resumo.Rmd

```
# Palavras finais
```

Terminamos um belo livro.

Por padrão, o **bookdown** mescla todos os arquivos Rmd pela ordem dos nomes dos arquivos, por exemplo, 01-intro.Rmd aparecerá antes de 02-literature.Rmd. Os nomes de arquivo que começam com um sublinhado são ignorados. Se existir um arquivo Rmd

chamado index.Rmd, ele sempre será tratado como o primeiro arquivo ao mesclar todos os arquivos Rmd. A razão para este tratamento especial é que o HTML arquivo index.html a ser gerado a partir de index.Rmd é geralmente o padrão arquivo de índice quando você visualiza um site, por exemplo, você está realmente navegando <http://yihui.org/index.html> quando você abre <http://yihui.org/>.

Você pode substituir o comportamento acima incluindo um arquivo de configuração chamado _bookdown.yml no diretório do livro. É um arquivo YAML (<https://en.wikipedia.org/wiki/YAML>), e os usuários do R Markdown devem estar familiarizados com este formato, pois ele também é usado para gravar os metadados em o início dos documentos R Markdown (você pode aprender mais sobre YAML na Seção B.2). Você pode usar um campo chamado rmd_files para definir sua própria lista e ordem de arquivos Rmd para o livro. Por exemplo,

```
rmd_files: ["index.Rmd", "abstract.Rmd", "intro.Rmd"]
```

Neste caso, o **bookdown** usará a lista de arquivos que você definiu neste campo YAML (index.Rmd será adicionado à lista se existir, e nomes de arquivos começando com sublinhados são sempre ignorados). Se você quer tanto HTML e saída LaTeX/PDF do livro, e use diferentes arquivos Rmd para saída HTML e LaTeX, você pode especificar esses arquivos para os dois formatos de saída separadamente, por exemplo,

```
rmd_files:
  html: ["index.Rmd", "abstract.Rmd", "intro.Rmd"]
  latex: ["abstract.Rmd", "intro.Rmd"]
```

Embora tenhamos falado sobre arquivos R Markdown, o capítulo os arquivos não precisam ser R Markdown. Eles podem ser arquivos Mark down simples (.md) e não precisam conter partes de código R. Você certamente pode usar o **bookdown** para compor romances ou poemas!

No momento, os principais formatos de saída que você pode usar incluem bookdown::pdf_book, bookdown::gitbook, bookdown::html_book e bookdown::epub_book. Há uma função bookdown::render_book() semelhante a rmarkdown::render(), mas foi projetado para renderizar vários documentos Rmd em um livro usando as funções de formato de saída. Você pode tanto

1.3 Uso

chame essa função diretamente da linha de comando ou clique nos botões relevantes no IDE do RStudio. Aqui estão alguns exemplos de linha de comando:

```
bookdown::render_book("foo.Rmd", "bookdown::gitbook")
bookdown::render_book("foo.Rmd", "bookdown::pdf_book")
bookdown::render_book("foo.Rmd", bookdown:: gitbook(lib_dir = "libs"))
bookdown::render_book("foo.Rmd", bookdown::pdf_book(keep_tex = TRUE))
```

Para usar render_book e as funções de formato de saída no RStudio IDE, você pode definir um campo YAML chamado site que recebe o valor book 4 e as down::bookdown_site, as funções de formato de saída podem ser usadas no campo de saída , por exemplo,

```
...
site: "bookdown::bookdown_site"
resultado:
  bookdown::gitbook:
    lib_dir: "book_assets"
  bookdown::pdf_book:
    keep_tex: sim
...
```

Em seguida, você pode clicar no botão Build Book no painel Build no RStudio para compilar os arquivos Rmd em um livro ou clicar no botão Knit na barra de ferramentas para visualizar o capítulo atual.

Mais opções de configuração de bookdown em **_bookdown.yml** são explicadas na Seção 4.4. Além dessas configurações, você também pode especificar algumas configurações relacionadas ao Pandoc nos metadados YAML do *primeiro* arquivo Rmd do livro, como título, autor, data do livro etc. Por exemplo:

```
...
title: "Criando um livro com R Markdown"
autor: "Yihui Xie"
```

⁴Esta função chama bookdown::render_book().

```

data: "r Sys.Date()"
site: "bookdown::bookdown_site"
resultado:
  bookdown::gitbook: default
  documentclass: livro
  bibliografia: ["book.bib", "packages.bib"]
  biblio-style: apalike link-
  citations: sim
  ...

```

1.4 Duas abordagens de renderização

Mesclar todos os capítulos em um arquivo Rmd e tricotá-lo é uma maneira de renderizar o livro em **bookdown**. Na verdade, existe outra maneira: você pode tricotar cada capítulo em uma sessão R separada e o **bookdown** mesclará a saída Markdown de todos os capítulos para renderizar o livro. Chamamos essas duas abordagens de “Merge and Knit” (MK) e “Knit and Merge” (KM), respectivamente. As diferenças entre eles podem parecer sutis, mas podem ser bastante importantes dependendo de seus casos de uso.

- A diferença mais significativa é que o MK executa *todos* os blocos de código em todos os capítulos na mesma sessão R, enquanto o KM usa sessões R separadas para capítulos individuais. Para MK, o estado da sessão R de capítulos anteriores é transferido para capítulos posteriores (por exemplo, objetos criados em capítulos anteriores estão disponíveis para capítulos posteriores, a menos que você os exclua deliberadamente); para KM, todos os capítulos são isolados uns dos outros.⁵ Se você quiser que cada capítulo seja compilado a partir de um estado limpo, use a abordagem KM. Pode ser muito complicado e difícil restaurar uma sessão R em execução para um estado completamente limpo se você usar a abordagem MK. Por exemplo, mesmo se você desanexar/descarregar pacotes carregados em um capítulo anterior, o R não limpará os métodos S3 registrados por esses pacotes.

⁵Claro, ninguém pode impedi-lo de escrever alguns arquivos em um capítulo, e lendo-os em outro capítulo. É difícil isolar esses tipos de efeitos colaterais.

- Como o **knitr** não permite rótulos de pedaços duplicados em um documento de origem, você precisa ter certeza de que não há rótulos duplicados em seus capítulos de livros quando usar a abordagem MK, caso contrário, o **knitr** sinalizará um erro ao tricotar o arquivo Rmd mesclado. Observe que isso significa que não deve haver rótulos duplicados em todo o livro.

A abordagem KM não requer rótulos duplicados em nenhum arquivo Rmd único.

- O KM não permite que os arquivos Rmd estejam em subdiretórios, mas o MK sim.

A abordagem padrão no **bookdown** é MK. Para mudar para KM, você pode usar o argumento `new_session = TRUE` ao chamar `render_book()`, ou definir `new_session: yes` no arquivo de configuração `_bookdown.yml`.

Você pode configurar a opção `book_filename` em `_bookdown.yml` para a abordagem KM, mas deve ser um nome de arquivo Markdown, por exemplo, `_main.md`, embora a extensão do nome do arquivo realmente não importe, e você pode até deixar de fora a extensão, por exemplo, apenas set `book_filename: _main`. Todas as outras configurações funcionam tanto para MK quanto para KM.

1.5 Algumas dicas

A tipografia sob a restrição de paginação (por exemplo, para saída LaTeX/PDF) pode ser um trabalho extremamente tedioso e demorado. Eu recomendo que você não veja sua saída PDF com frequência, já que na maioria das vezes é muito improvável que você fique satisfeito: o texto pode transbordar para a margem da página, as figuras podem flutuar muito longe e assim por diante. Não tente fazer as coisas parecerem certas *imediatamente*, porque você pode se decepcionar repetidamente enquanto continua revisando o livro, e as coisas podem ficar confusas novamente, mesmo que você tenha feito apenas algumas pequenas alterações (consulte <http://bit.ly/tbrLtx> para uma bela ilustração).

Se você quiser visualizar o livro, visualize a saída HTML. Trabalhe na versão em PDF depois de terminar o conteúdo do livro e tenha certeza de que nenhuma revisão importante será necessária.

Se determinados fragmentos de código em seus documentos R Markdown forem demorados para serem executados, você poderá armazená-los em cache adicionando a opção `fragment`

cache = TRUE no cabeçalho do fragmento, e é recomendável rotular esses fragmentos de código também, por exemplo,

```
```{r importante-computação, cache=TRUE}
```

No Capítulo 5, falaremos sobre como visualizar rapidamente um livro enquanto você edita arquivos . Resumindo, você pode usar a função `preview_chapter()` para renderizar um único capítulo em vez de todo o livro. A função `serve_book()` facilita a visualização ao vivo de páginas de livros HTML: sempre que você modifica um arquivo Rmd, o livro pode ser recompilado e o navegador pode ser atualizado automaticamente de acordo.

# 2

---

## Componentes

---

Este capítulo demonstra a sintaxe de componentes comuns de um livro escrito em **bookdown**, incluindo pedaços de código, figuras, tabelas, citações, teoremas matemáticos e equações. A abordagem é baseada no Pan doc, então começamos com a sintaxe do sabor de Markdown do Pandoc.

---

### 2.1 Sintaxe de redução

Nesta seção, fazemos uma breve introdução ao Pandoc's Mark down. Os leitores familiarizados com o Markdown podem pular esta seção.

A sintaxe abrangente do Markdown do Pandoc pode ser encontrada no Site da Pandoc <http://pandoc.org>.

#### 2.1.1 Formatação em linha

Você pode tornar o texto em *italico* cercando-o com sublinhados ou asteriscos, por exemplo, `_text_` ou `*text*`. Para texto em **negrito**, use duas pontuações abaixo (`__text__`) ou asteriscos (`**text**`). Texto cercado por `~` will ser convertido em um subscrito (por exemplo, `H~2~SO~4~` renderiza  $H_2SO_4$ ), e da mesma forma, dois acentos circunflexos (`^`) produzem um sobreescrito (por exemplo, `Fe^2+^` renderiza  $Fe^{2+}$ ). Para marcar o texto como código embutido, use um par de acentos graves, por exemplo, 'código'.<sup>1</sup> Maiúsculas podem ser produzidas pela extensão da tag HTML, por exemplo, `<span style="font-variant:small-caps;">Small Caps renderiza SMALL CAPS. Os links são criados usando [texto](link), por exemplo, [RStudio](https://www.rstudio.com), e a sintaxe para imagens é semelhante: basta adicionar um ponto de exclamação, por exemplo, ![texto alternativo ou imagem]`

---

<sup>1</sup>Para incluir acentos graves literais, use mais acentos graves fora, por exemplo, você pode usar dois backticks para preservar um backtick dentro de: `` `code` `` .

tle](caminho/para/imagem). As notas de rodapé são colocadas entre colchetes após um acento circunflexo ^[], por exemplo, ^[Esta é uma nota de rodapé.]. Falaremos sobre citações na Seção 2.8.

### 2.1.2 Elementos de nível de bloco

Os cabeçalhos de seção podem ser escritos após vários sinais de libra, por exemplo,

```
Cabeçalho de primeiro nível

Cabeçalho de segundo nível

Cabeçalho de terceiro nível
```

Se você não quiser que um determinado título seja numerado, você pode adicionar {-} após o título, por exemplo,

```
Prefácio {-}
```

Os itens de lista não ordenados começam com \*, - ou +, e você pode aninhar uma lista dentro de outra lista recuando a sublistas por quatro espaços, por exemplo,

```
- um artigo
- um artigo
- um artigo
 - um artigo
 - um artigo
```

A saída é:

```
• um item •
um item • um
item – um
 item – um item
```

Itens de lista ordenada começam com números (a regra para listas aninhadas é a mesma acima), por exemplo,

## 2.1 Sintaxe de redução

1. o primeiro item
2. o segundo item
3. o terceiro item

A saída não parece muito diferente com o Markdown fonte:

1. o primeiro item
2. o segundo item
3. o terceiro item

Blockquotes são escritos após >, por exemplo,

```
> "Eu desaprovo completamente os duelos. Se um homem me desafiasse, eu o pegaria pela mão com bondade
e perdão e o levaria
para um lugar tranquilo e matá-lo."
>
> --- Mark Twain
```

A saída real (personalizamos o estilo para blockquotes neste livro):

---

"Eu desaprovo completamente os duelos. Se um homem me desafiasse, eu o pegaria  
pela mão com bondade e perdão e o levaria a um lugar tranquilo e o mataria."

— Mark Twain

---

Blocos de código simples podem ser escritos após três ou mais acentos graves, e você também pode recuar os blocos por quatro espaços, por exemplo,

```

Este texto é exibido literalmente / pré-formatado

Ou recue por quatro espaços:

Este texto é exibido literalmente / pré-formatado

2.1.3 Expressões matemáticas

Equações de LaTeX inline podem ser escritas em um par de cifrões usando o Sintaxe do LaTeX, por exemplo, $f(k) = \binom{n}{k} p^k (1-p)^{n-k}$ (real saída: $() = () (1 \ddot{\gamma}) \ddot{\gamma} ()$); expressões matemáticas do visor

estilo pode ser escrito em um par de cifrões duplos, por exemplo, $\$f(k) = \binom{n}{k} p^k (1-p)^{n-k}\$$, e a saída se parece com isso:

$$() = () (1 \ddot{\gamma}) \ddot{\gamma} ()$$

Você também pode usar ambientes matemáticos dentro de \$\$ ou \$\$\$, por exemplo,

```
$$\begin{array}{ccc}
x_{11} & x_{12} & x_{13} \\
x_{21} & x_{22} & x_{23}
\end{array}$$
```

| | | |
|----|----|----|
| 11 | 12 | 13 |
| 21 | 22 | 23 |

```
$$X = \begin{bmatrix} 1 & x_{12} & x_{13} \\
1 & x_{22} & x_{23} \\
1 & x_{32} & x_{33}
\end{bmatrix}$$
```

$$= \begin{matrix} & 1 \\ \ddot{\gamma} & 1 & 1\ddot{\gamma} \\ & \ddot{\gamma} & 2\ddot{\gamma} \\ & & 3 \end{matrix}$$

```
$$\Theta = \begin{pmatrix} \alpha & \beta \\ \gamma & \delta \end{pmatrix}
```

$$\ddot{\gamma} = ()$$

```
$$\begin{vmatrix} a & b \\ c & d \end{vmatrix} = ad - bc
```

$$\ddot{\gamma} = \ddot{\gamma}$$

2.2 Extensões Markdown por bookdown

Embora o Markdown do Pandoc seja muito mais rico do que a sintaxe original do Markdown, ainda falta uma série de coisas que podemos precisar para escrita acadêmica. Por exemplo, ele suporta equações matemáticas, mas você não pode numerar e referenciar equações em HTML ou EPUB de várias páginas resultado. Fornecemos algumas extensões de Markdown no **bookdown** para preencher as lacunas.

2.2.1 Número e equações de referência

Para numerar e se referir a equações, coloque-as nos ambientes de equação e atribua rótulos a elas usando a sintaxe (#eq:label), por exemplo,

```
\begin{equação}
\left( k \right) = \binom{n}{k} p^k \left( 1-p \right)^{n-k}
\left( \#eq:binom \right)
\end{equação}
```

Ele renderiza a equação abaixo:

$$\left(k \right) = \binom{n}{k} p^k \left(1-p \right)^{n-k} \quad (2.1)$$

Você pode se referir a ele usando `\@ref(eq:binom)`, por exemplo, veja a Equação (2.1).



Os rótulos de equação devem começar com o prefixo `eq:` no **bookdown**. Todas as etiquetas no **bookdown** devem conter apenas caracteres alfanuméricos, :, -, , e/ou /. Referências de equação funcionam melhor para saída LaTeX/PDF, e eles não são bem suportados na saída do Word ou e-books. Por Saída HTML, **bookdown** só pode numerar as equações com rótulos. Certifique-se de que as equações sem rótulos não sejam numeradas usando o ambiente `equação*` ou adicionando `\nonumber` ou `\notag` às suas equações. As mesmas regras se aplicam a outros ambientes matemáticos, como `eqnarray`, `reunir`, `alinear` e assim por diante (por exemplo, você pode usar o comando `align*` meio Ambiente).

Demonstramos mais alguns ambientes de equações matemáticas abaixo. Aqui é uma equação não numerada usando o ambiente `equação*`:

```
\begin{equação*}
\frac{d}{dx} \left( \int_a^x f(u) du \right) = f(x)
\end{equação*}
```

$$\frac{d}{dx} \left(\int_a^x f(u) du \right) = f(x)$$

Abaixo está um ambiente de alinhamento (2.2):

2.2 Extensões Markdown por bookdown

```
\begin{align}
g(X_{\{n\}}) &= g(\theta) + g'(\tilde{\theta})(X_{\{n\}} - \theta) \notag \\
\sqrt{n}[g(X_{\{n\}}) - g(\theta)] &= g'(\tilde{\theta}) \notag \\
\sqrt{n}[X_{\{n\}} - \theta] \quad (\#eq:align)
\end{aligned}
```

$$\begin{aligned} (\) &= (\) + \frac{\partial}{\partial \theta} (\) \notag \\
\bar{y} [\bar{y}] &= (\) \bar{y} [\bar{y}] \quad (2.2) \end{aligned}$$

Você pode usar o ambiente de divisão dentro da equação para que todas as linhas compartilhem o mesmo número (2.3). Por padrão, cada linha no ambiente de alinhamento receberá um número de equação. Suprimimos o número da primeira linha no exemplo anterior usando `\notag`. Neste exemplo, todo o ambiente de divisão recebeu um único número.

```
\begin{equation}
\begin{aligned}
\mathrm{Var}(\hat{\beta}) &= \mathrm{Var}((X'X)^{-1}X'y) \\
&= (X'X)^{-1}X'\mathrm{Var}(y)(X'X)^{-1}X' \\
&= (X'X)^{-1}X'\mathrm{Var}(y)X(X'X)^{-1} \\
&= (X'X)^{-1}\sigma^2 I(X(X'X)^{-1}) \\
&= (X'X)^{-1}\sigma^2
\end{aligned}
\quad (\#eq:var-beta)
\end{equation}
```

$$\begin{aligned} \mathrm{Var}(\) &= \mathrm{Var}((\)) \notag \\
&= (\) \mathrm{Var}((\)) (\) \notag \\
&= (\) \mathrm{Var}((\)) (\) = (\) \mathrm{Var}((\)) = \\
&= (\) \mathrm{Var}((\))
\end{aligned} \quad (2.3)$$

2.2.2 Teoremas e provas

Teoremas e demonstrações são comumente usados em artigos e livros de matemática. No entanto, por favor, não se deixe enganar pelos nomes: um “teorema” é apenas um ambiente numerado/rotulado, e não precisa ser um teorema matemático (por exemplo, pode ser um exemplo irrelevante para a matemática). Da mesma forma, uma “prova” é um ambiente não numerado.

Nesta seção, sempre usamos os significados *gerais* de um “teorema” e “prova”, a menos que explicitamente declarado.

Em **bookdown**, os tipos de ambientes de teoremas suportados estão na Tabela 2.1. Para escrever um teorema, você pode usar a sintaxe abaixo:

```
::: {.teorema}
Este é um ambiente `teorema` que pode conter **qualquer** sintaxe
_Markdown_.
:::
```

Essa sintaxe é baseada nos blocos Div protegidos do Pandoc2 e já pode ser usada em qualquer documento R Markdown para escrever blocos personalizados.³ **Book down** oferece apenas tratamento especial para ambientes de teorema e prova. Como isso usa a sintaxe do Markdown do Pandoc, você pode escrever qualquer texto Markdown válido dentro do bloco.

Para escrever outros ambientes de teoremas, substitua `::: {.theorem}` por outros nomes de ambiente na Tabela 2.1, por exemplo, `::: {.lemma}`.

Um teorema pode ter um atributo de nome para que seu nome seja impresso. Por exemplo,

```
::: {.theorem name="teorema de Pitágoras"}
Para um triângulo retângulo, se $c$ denota o comprimento da hipotenusa
e $a$ e $b$ denotam os comprimentos dos outros dois lados, temos
$$a^2 + b^2 = c^2$$
:::
```

² <https://pandoc.org/MANUAL.html#divs-and-spans>

³ <https://bookdown.org/yihui/rmarkdown-cookbook/custom-blocks.html>

2.2 Extensões Markdown por bookdown

TABELA 2.1: Ambientes de teoremas em **bookdown**.

| Prefixo da etiqueta do nome impresso do ambiente | | |
|--|-----------------------|-----|
| teorema | Teorema | thm |
| lema | Lema | lem |
| corolário | Proposição | cor |
| Corolária | Proposição Conjectura | prp |
| | Conjectura | cnj |
| hipótese de | Definição | def |
| exercício de | Exemplo | exm |
| exemplo de | Exercício | ex |
| definição | Hipótese | hip |

Se você quiser se referir a um teorema, você deve rotulá-lo. A etiqueta pode ser fornecido como um ID para o bloco do formulário `#label`. Por exemplo,

```
::: {.teorema #foo}
Um teorema rotulado aqui.
:::
```

Depois de rotular um teorema, você pode consultá-lo usando a sintaxe `\@ref(prefixo:rótulo)`. Consulte a coluna Prefixo do rótulo na Tabela 2.1 para o valor do prefixo para cada ambiente. Por exemplo, temos uma etiqueta e nomeado teorema abaixo, e `\@ref(thm:pyth)` nos dá seu teorema número 2.1:

```
::: {.teorema #pyth name="teorema de Pitágoras"}
Para um triângulo retângulo, se  $c$  denota o comprimento da hipotenusa
e  $a$  e  $b$  denotam os comprimentos dos outros dois lados, temos
```

```
$$a^2 + b^2 = c^2$$
:::
```

Teorema 2.1 (Teorema de Pitágoras). *Para um triângulo retângulo, se denota o*

comprimento da hipotenusa e e denotam os comprimentos dos outros dois lados, temos

$$2^2 + 2^2 = 2^2$$

Os ambientes de prova suportados atualmente são proof, remark e solution. A sintaxe é semelhante aos ambientes de teorema e os ambientes de prova também podem ser nomeados usando o atributo name . A única diferença é que, como eles não são numerados, você não pode referenciá-los, mesmo se você fornecer um ID para um ambiente de prova.

Tentamos fazer com que todos esses ambientes de teorema e prova trabalhe fora da caixa, não importa se sua saída é PDF ou HTML. Se você é um especialista em LaTeX ou HTML, você pode querer personalizar o estilo desses ambientes de qualquer maneira (veja o Capítulo 4). Costumização em HTML é fácil com CSS, e cada ambiente é incluído em `<div></div>` com a classe CSS sendo o nome do ambiente, por exemplo, `<div class="lema"></div>`. Para saída LaTeX, predefinimos o estilo ser definição para definição de ambientes , exemplo, exercício e hipótese, e observação para ambientes prova e observação. Todos os outros ambientes utilizam o estilo simples . A definição do estilo é feita através do comando `\theoremstyle{}` do pacote **amsthm** . Se você não quer as definições de teorema padrão a serem adicionadas automaticamente pelo **bookdown**, você pode definir `options(bookdown.theorem.preamble = FALSE)`. este pode ser útil, por exemplo, para evitar conflitos em documentos únicos (Seção 3.4) usando o formato de saída `bookdown::pdf_bookwith` a `base_format` que já incluiu definições de **amsmath** .

Os teoremas são numerados por capítulos por padrão. Se não houver capítulos em seu documento, eles serão numerados por seções. Se todo o documento não é numerado (a opção de formato de saída número `ber_sections = FALSE`), todos os teoremas são numerados sequencialmente de 1, 2, ..., N. LaTeX suporta numerar um ambiente de teorema após outro, por exemplo, deixe teoremas e lemas compartilharem o mesmo contador. não suportado para saída HTML/EPUB em **bookdown**. Você pode mudar o esquema de numeração no preâmbulo do LaTeX definindo seu próprio ambientes de teoremas, por exemplo,

2.2 Extensões Markdown por bookdown

```
\novotheorema{teorema}{teorema}
\novotheorema{lema}[teorema][lema]
```

Quando **bookdown** detecta `\newtheorem{theorem}` em seu preâmbulo LaTeX, ele não escreverá suas definições de teorema padrão, o que significa você tem que definir todos os ambientes de teoremas por si mesmo. Para o bem de simplicidade e consistência, não recomendamos que você faça isso. Pode ser confuso quando seu Teorema 18 em PDF se torna Teorema 2.4 em HTML.

Abaixo mostramos mais exemplos⁴ dos ambientes de teorema e prova, para que você possa ver os estilos padrão no **bookdown**.

Definição 2.1. A função característica de uma variável aleatória é definido por

$$() = E [\dots] , \quad \ddot{\text{y}} \ddot{\text{y}}$$

Exemplo 2.1. Derivamos a função característica de $\mathcal{Y} \sim U(0, 1)$ com a função de densidade de probabilidade $() = 1_{[0,1]}$.

⁴Alguns exemplos são adaptados da página da Wikipédia [https://en.wikipedia.org/wiki/Func%C3%A3o_caracter%C3%ADstica_\(teoria_probabilidade\)](https://en.wikipedia.org/wiki/Func%C3%A3o_caracter%C3%ADstica_(teoria_probabilidade))

$$() = E [\quad]$$

$$= \ddot{y} () = \ddot{y} 1$$

0

$$= \ddot{y} 1_0 (\cos() + \sin())$$

$$\begin{aligned} &= (\overline{\text{pecado}()} - \overline{\cos()}) \ddot{y} 1_0 \\ &= \frac{\text{pecado}()}{\ddot{y} (\cos() \ddot{y} 1)} \\ &= \frac{\text{pecado}()}{\ddot{y} 1} + \frac{\cos() \ddot{y} 1}{\ddot{y} 1} \\ &= \frac{\ddot{y} 1}{\ddot{y} 1} \end{aligned}$$

Observe que usamos o fato $= \cos() + \sin()$ duas vezes.

Lema 2.1. Para quaisquer duas distribuições de $1, 2, \dots$, ambos têm o mesmo probabilidade de variáveis aleatórias se e somente se

$$_1() = _2()$$

Teorema 2.2. Se são variáveis aleatórias independentes e $1, \dots$, algumas constantes, então a função característica da combinação linear $= \ddot{y} = \ddot{y} \ddot{y} 1$ é

$$() = \ddot{y} \quad () = \quad _1(_1) \ddot{y} (_2)$$

Proposta 2.1. A distribuição da soma de Poisson aleatório independente variáveis \ddot{y} Ervilhas($1, 2, \dots$), $\ddot{y} = \ddot{y} = \ddot{y} 1$, is Pois($\ddot{y} = \ddot{y} 1$).

Prova. A função característica de \ddot{y} Pois(1) é $() = (-1)^n$. Sabemos pelo Teorema 2.2 que

$$() = \ddot{y} \quad ()$$

$$= \ddot{y} \quad (-1)$$

$$= \ddot{y} \quad (-1)$$

Esta é a função característica de uma variável aleatória de Poisson com o parâmetro $= \ddot{y}$. Do Lema 2.1, sabemos que a distribuição de é $\text{Pois}(\ddot{y})$. \square

Observação. Em alguns casos, é muito conveniente e fácil descobrir o distribuição da soma de variáveis aleatórias independentes usando funções características.

Corolário 2.1. A função característica da soma de duas variáveis aleatórias independentes é o produto de funções características de e^x

, ou seja,

$$e^{x_1 + x_2} = e^{x_1} e^{x_2}$$

Exercício 2.1 (Função Característica da Média Amostra). Deixe $\bar{y} = \frac{1}{n} \sum_{i=1}^n y_i$ seja a média amostral de valores independentes e idênticos variáveis aleatórias distribuídas, cada uma com função . característica Calcule a função característica de \bar{y} .

Solução. Aplicando o Teorema 2.2, temos

$$\ddot{y}() = \ddot{y} \quad () = [\quad ()] .$$

Hipótese 2.1 (hipótese de Riemann). A função Zeta de Riemann é definido como

$$\zeta(s) = \sum_{n=1}^{\infty} \frac{1}{n^s}$$

para valores complexos de s que converge quando a parte real de

é maior que 1. A hipótese de Riemann é que a função zeta de Riemann tem seus zeros apenas nos inteiros pares negativos e números complexos com parte real 1/2.

2.2.2.1 Uma nota sobre a sintaxe antiga

Para versões mais antigas do **bookdown** (antes da v0.21), um ambiente de teorema pode ser escrito assim:

```
```{teorema pyth, name="teorema de Pitágoras"}
Para um triângulo retângulo, se c denota o comprimento da hipotenusa
e a e b denotam os comprimentos dos outros dois lados, temos

$$a^2 + b^2 = c^2$$
```

Essa sintaxe ainda funciona, mas não a recomendamos, pois o novo syn tax permite que você escreva conteúdo mais rico e tenha uma implementação mais limpa.

Essa conversão entre as duas sintaxes é direta. O teorema acima pode ser reescrito da seguinte forma:

```
::: {.teorema #pyth name="teorema de Pitágoras"}
Para um triângulo retângulo, se c denota o comprimento da hipotenusa e a e b denotam os comprimentos dos outros dois lados, temos

$$a^2 + b^2 = c^2$$
:::
```

Você pode usar a função auxiliar `bookdown::fence_theorems()` para converter um arquivo inteiro ou um pedaço de texto. Esta é uma operação única. Tentamos fazer a conversão da sintaxe antiga para a nova com segurança, mas podemos ter perdido alguns casos extremos. Para garantir que você não sobrescreva o arquivo de entrada acidentalmente, você pode gravar a fonte convertida em um novo arquivo, por exemplo,

```
bookdown::fence_theorems("01-intro.Rmd", output = "01-intro-new.Rmd")
```

Em seguida, verifique novamente o conteúdo de 01-intro-new.Rmd. Usar `output = NULL` imprimirá o resultado da conversão no console do R e é outra maneira de verificar a conversão. Se você estiver usando uma ferramenta de versão de controle, poderá definir a saída para ser igual à entrada, pois deve ser seguro e fácil reverter a alteração se algo der errado.

### 2.2.3 Cabeçalhos especiais

Existem alguns tipos especiais de cabeçalhos de primeiro nível que serão processados de forma diferente no **bookdown**. O primeiro tipo é um cabeçalho não numerado que começa com o token (PART). Esse tipo de cabeçalho é traduzido em títulos de partes. Se você estiver familiarizado com o LaTeX, isso significa basicamente `\part{}`. Quando seu livro tem um grande número de capítulos, você pode organizá-los em partes, por exemplo,

```
(PARTE) Parte I {-}
```

```
Capítulo um
```

```
Capítulo dois
```

```
(PARTE) Parte II {-}
```

```
Capítulo três
```

Um título de parte deve ser escrito logo antes do título do primeiro capítulo desta parte, ambos no mesmo documento. Você pode usar (PART<sup>\*</sup>) (a barra invertida antes de \* é obrigatória) em vez de (PART) se um título de parte não deve ser numerado.

O segundo tipo é um cabeçalho não numerado que começa com (APÊNDICE), indicando que todos os capítulos após este cabeçalho são apêndices, por exemplo,

```
Capítulo um
```

```
Capítulo dois
```

```
(APÊNDICE) Apêndice {-}
```

```
Apêndice A
```

```
Apêndice B
```

O estilo de numeração dos apêndices será alterado automaticamente na saída LaTeX/PDF e HTML (geralmente na forma A, A.1, A.2, B, B.1, ...). Esse recurso não está disponível para e-books ou saída do Word.

#### 2.2.4 Referências de texto

Você pode atribuir algum texto a um rótulo e fazer referência ao texto usando o rótulo em outro lugar do documento. Isso pode ser particularmente útil para legendas longas de figuras/tabelas (Seção 2.4 e 2.5), caso em que você normalmente terá que escrever toda a cadeia de caracteres no cabeçalho do bloco (por exemplo, `fig.cap = "Uma legenda de figura longa longa. "`) ou seu código R (por exemplo, `kable(caption = "A long long table caption.")`). Também é útil quando essas legendas contêm caracteres HTML ou LaTeX especiais, por exemplo, se a legenda da figura contém um sublinhado, ela funciona na saída HTML, mas pode não funcionar na saída LaTeX porque o sublinhado deve ser escapado no LaTeX.

A sintaxe para uma referência de texto é `(ref:label) text`, onde rótulo é um rótulo exclusivo<sup>5</sup> em todo o documento para texto. Deve estar em um parágrafo separado com linhas vazias acima e abaixo dele. O parágrafo não deve ser dividido em várias linhas e não deve terminar com um espaço em branco. Por exemplo,

```
(ref:foo) Defina uma referência de texto **aqui**.
```

Então você pode usar `(ref:foo)` em suas legendas de figuras/tabelas. O texto pode conter qualquer coisa que o Markdown suporte, desde que seja um único parágrafo. Aqui está um exemplo completo:

---

<sup>5</sup>Você pode considerar usar os rótulos de trecho de código.

## 2.4 código R

Um parágrafo normal.

(ref:foo) Um gráfico de dispersão dos dados `cars` usando gráficos \*\*base\*\* R.

```
```{r foo, fig.cap=(ref:foo)}
plot(cars) # um gráfico de dispersão
```

As referências de texto podem ser usadas em qualquer lugar do documento (não se limitando a legendas de figuras). Também pode ser útil se você quiser reutilizar um fragmento de texto em vários lugares.

2.3 código R

Existem dois tipos de código R em documentos R Markdown/knitr: fragmentos de código R e código R embutido. A sintaxe para este último é `r R_CODE`, e pode ser embutido em linha com outros elementos do documento. Os fragmentos de código R parecem blocos de código simples, mas têm {r} após os três acentos graves e (opcionalmente) opções de fragmento dentro de {}, por exemplo,

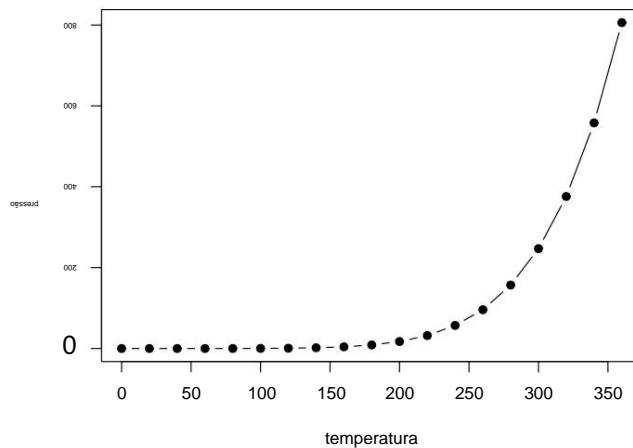
```
```{r chunk-label, echo = FALSE, fig.cap = 'Uma legenda de figura.'}
1 + 1
rnorm(10) # 10 números aleatórios
plot(dist ~ velocidade, carros) # um gráfico de dispersão
```

Para saber mais sobre as opções **do knitr** chunk, consulte [Xie \(2015\)](#) ou a página da web <http://yihui.org/knitr/options>. Para livros, código R adicional pode ser executado antes/depois de cada capítulo; veja before\_chapter\_script e after\_chapter\_script na Seção 4.4.

## 2.4 Figuras

Por padrão, as figuras não têm legendas na saída gerada pelo **knitr**, o que significa que eles serão colocados onde quer que tenham sido gerados no código R. Abaixo está um exemplo.

```
par(mar = c(4, 4, 0, 1, 0, 1))
plot(pressão, pch = 19, tipo = "b")
```



A desvantagem de compor figuras desta forma é que quando há  
não há espaço suficiente na página atual para colocar uma figura, ela pode atingir a  
parte inferior da página (excedendo a margem da página) ou  
ser empurrado para a próxima página, deixando uma grande margem branca na parte  
inferior da página atual. É basicamente por isso que existem “ambientes flutuantes” no  
LaTeX: elementos que não podem ser divididos em várias páginas  
(como figuras) são colocados em ambientes flutuantes, para que possam flutuar para um  
página que tem espaço suficiente para mantê-los. Há também uma desvantagem  
de flutuar as coisas para frente ou para trás, no entanto. Ou seja, os leitores podem  
tem que pular para uma página diferente para encontrar a figura mencionada no  
pagina atual. Isso é simplesmente uma consequência natural de ter que digitar coisas  
definidas em várias páginas de tamanhos fixos. Este problema não existe em  
HTML, no entanto, uma vez que tudo pode ser colocado continuamente em um

## 2.4 Figuras

29

página única (presumivelmente com altura infinita) e não há necessidade de dividir nada em várias páginas do mesmo tamanho de página.

Se atribuirmos uma legenda de figura a um pedaço de código por meio da opção de pedaço `fig.cap`, os gráficos R serão colocados em ambientes de figura, que serão rotulados e numerados automaticamente e também podem ser referenciados.

O rótulo de um ambiente de figura é gerado a partir do rótulo do pedaço de código, por exemplo, se o rótulo do pedaço for `foo`, o rótulo da figura será `fig:foo` (o prefixo `fig:` é adicionado antes de `foo`). Para referenciar uma figura, use a sintaxe `\@ref(label)`, <sup>6</sup> onde `label` é o rótulo da figura, por exemplo, `fig:foo`.

Para tirar proveito da formatação Markdown na legenda da figura, você precisará usar referências de texto (consulte a Seção 2.2.4). Por exemplo, uma legenda de figura que contém `_italic text_` não funcionará quando o formato de saída for LaTeX/PDF, pois o sublinhado é um caractere especial no LaTeX, mas se você usar referências de texto, `_italic text_` será traduzido para código LaTeX quando o saída é LaTeX.



Se você quiser fazer referência cruzada a figuras ou tabelas geradas a partir de um trecho de código, certifique-se de que o rótulo do trecho contenha apenas caracteres alfanuméricos (az, AZ, 0-9), barras (/) ou traços (-).

A opção `chunk fig.asp` pode ser usada para definir a relação de aspecto dos gráficos, ou seja, a relação altura/largura da figura. Se a largura da figura for 6 polegadas (`fig.width = 6`) e `fig.asp = 0,7`, a altura da figura será calculada automaticamente a partir de `fig.width * fig.asp = 6 * 0,7 = 4,2`. A Figura 2.1 é um exemplo usando as opções de `chunk fig.asp = 0,7`, `fig.width = 6` e `fig.align = 'center'`, geradas a partir do código abaixo:

```
par(mar = c(4, 4, 0, 1, 0, 1))
plot(pressão, pch = 19, tipo = "b")
```

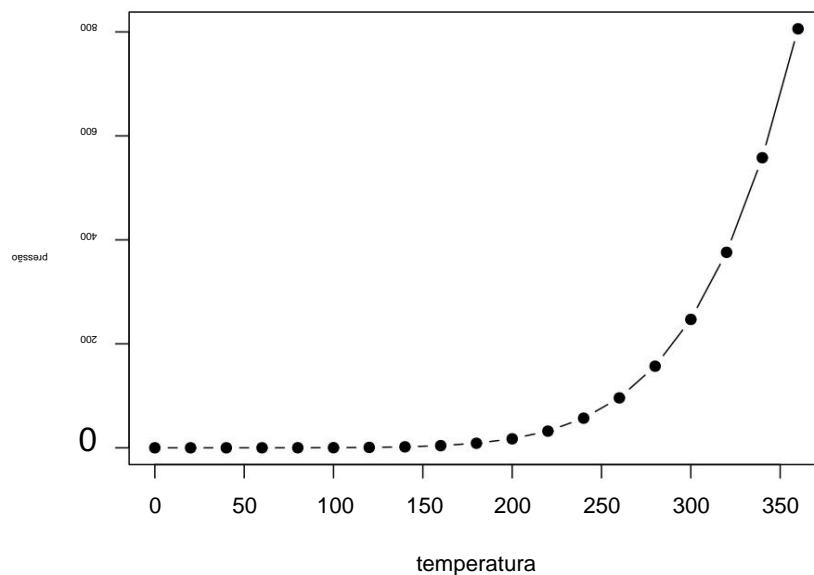
O tamanho real de um gráfico é determinado pelas opções do bloco `fig.width` e `fig.height` (o tamanho do gráfico gerado a partir de um dispositivo gráfico), e podemos especificar o tamanho de saída dos gráficos por meio das opções do bloco

---

<sup>6</sup>Não se esqueça da barra invertida inicial! E também observe os parênteses () após `ref`; não são chaves {}.

30

2 Componentes



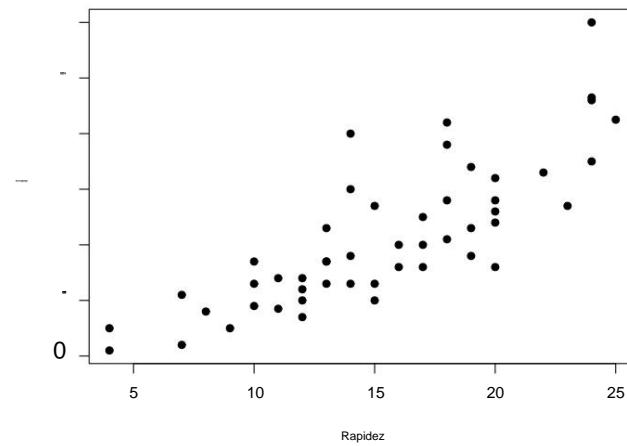
**FIGURA 2.1:** Um exemplo de figura com a proporção especificada, largura, e alinhamento.

out.width e out.height. O valor possível dessas duas opções depende do formato de saída do documento. Por exemplo, fora.largura = '30%' é um valor válido para saída HTML, mas não para saída LaTeX/PDF. No entanto, o **knitr** converterá automaticamente um valor percentual para out.width da forma x% para (x / 100) \linewidth, por exemplo, out.width = '70%' será tratado como .7\linewidth quando o formato de saída for LaTeX. Este torna possível especificar uma largura relativa de um gráfico de forma consistente maneiras. A Figura 2.2 é um exemplo de out.width = 70%.

```
par(mar = c(4, 4, 0.1, 0.1))
plot(carros, pch = 19)
```

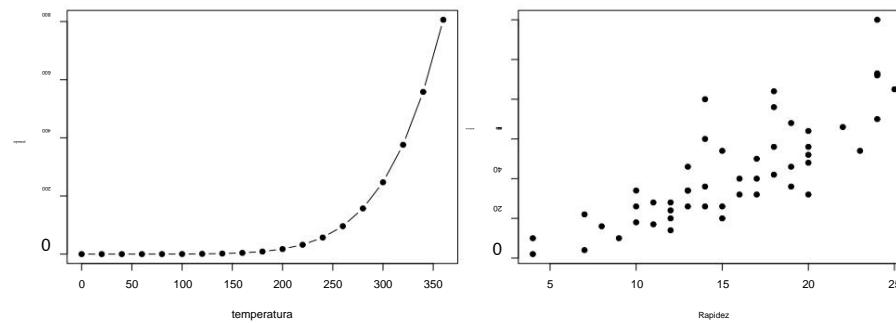
Se você quiser colocar vários gráficos em um ambiente de figura, você deve usar a opção chunk.fig.show = 'hold' para armazenar vários gráficos de um pedaço de código e incluí-los em um ambiente. Você também pode colocar plotagens lado a lado se a soma da largura de todas as parcelas for menor ou igual à largura da linha atual. Por exemplo, se duas parcelas têm o mesmo largura 50%, eles serão colocados lado a lado. Da mesma forma, você pode especificar

## 2.4 Figuras

**FIGURA 2.2:** Exemplo de figura com largura relativa de 70%.

`out.width = '33%'` para organizar três gráficos em uma linha. A Figura 2.3 é uma exemplo de duas parcelas, cada uma com uma largura de 50%.

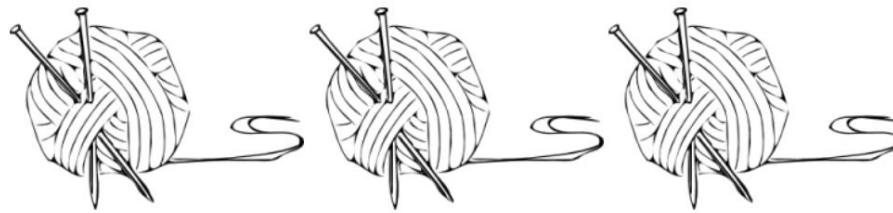
```
par(mar = c(4, 4, 0.1, 0.1))
plot(pressão, pch = 19, tipo = "b")
plot(carros, pch = 19)
```

**FIGURA 2.3:** Duas parcelas colocadas lado a lado.

Às vezes você pode ter certas imagens que não são geradas a partir de código R, e você pode incluí-los no R Markdown através da função `knitr::include_graphics()`. A Figura 2.4 é um exemplo de três **logotipos** de tricô incluídos em um ambiente de figura. Você pode passar um ou vários caminhos de imagem para a função `include_graphics()` e todas as opções de partes

que se aplicam a plotagens R normais também se aplicam a essas imagens, por exemplo, você pode usar `out.width = '33%'` para definir as larguras dessas imagens na saída documento.

```
knitr::include_graphics(rep("images/knit-logo.png", 3))
```



**FIGURA 2.4:** Três logotipos do knitr incluídos no documento de um arquivo de imagem PNG externo.

Existem algumas vantagens de usar `include_graphics()`:

1. Você não precisa se preocupar com o formato de saída do documento, por exemplo, quando o formato de saída é LaTeX, você pode ter que usar o comando LaTeX `\includegraphics{}` para incluir uma imagem, e quando o formato de saída é Markdown, você deve usar `![]()`. A função `include_graphics()` no **knitr** cuida de esses detalhes automaticamente.
2. A sintaxe para controlar os atributos da imagem é a mesma quando as imagens são geradas a partir do código R, por exemplo, opções de blocos `fig.cap`, `out.width` e `fig.show` ainda têm os mesmos significados.
3. `include_graphics()` pode ser inteligente o suficiente para usar gráficos PDF automaticamente quando o formato de saída for LaTeX e o Existem arquivos gráficos PDF, por exemplo, um caminho de imagem `foo/bar.png` pode ser automaticamente substituído por `foo/bar.pdf` se este existir. Imagens PDF geralmente têm qualidades melhores do que imagens raster na saída LaTeX/PDF. Para usar este recurso, defina o argumento `auto_pdf = TRUE`, ou defina as opções de opções globais (`knitr.graphics.auto_pdf = TRUE`) para habilitar este recurso globalmente em uma sessão R.
4. Você pode dimensionar facilmente essas imagens proporcionalmente usando o

mesma proporção. Isso pode ser feito através do argumento dpi (pontos por polegada), que obtém o valor da opção de bloco dpi por defeito. Se for um valor numérico e a opção de bloco out.width não estiver definido, a largura de saída de uma imagem será sua real largura (em pixels) dividida por dpi, e a unidade será polegadas. Por exemplo, para uma imagem com o tamanho 672 x 480, sua saída a largura será de 7 polegadas (7 polegadas) quando dpi = 96. Este recurso requer que o pacote **png** e/ou **jpeg** seja instalado. Você sempre pode substituir o cálculo automático da largura em polegadas por fornecendo um valor não NULL para a opção de bloco out.width, ou use `include_graphics(dpi = NA)`.

## 2.5 Tabelas

Por enquanto, a maneira mais conveniente de gerar uma tabela é a função `knitr::kable()`, porque existem alguns truques internos no **knitr** para fazer funciona com **bookdown** e os usuários não precisam saber nada sobre esses detalhes de implementação. Explicaremos como usar outros pacotes e funções posteriormente nesta seção.

Assim como as figuras, as tabelas com legendas também serão numeradas e poderão ser referenciadas. A função `kable()` irá gerar automaticamente um rótulo para um ambiente de tabela, que é a guia de prefixo: mais o rótulo do pedaço. Por exemplo, o rótulo da tabela para um pedaço de código com o rótulo `foo` será `tab:foo`, e ainda podemos usar a sintaxe `\@ref(label)` para referenciar a tabela. A Tabela 2.2 é um exemplo simples.

```
tricotar::kable(
 head(mtcars[, 1:8], 10), booktabs = TRUE,
 caption = 'Uma tabela das primeiras 10 linhas dos dados mtcars.'
)
```

Se você quiser colocar várias tabelas em um único ambiente de tabela, envolva os objetos de dados (geralmente quadros de dados em R) em uma lista. Consulte a Tabela 2.3 para

**TABELA 2.2:** Uma tabela das primeiras 10 linhas dos dados mtcars.

	mpg	cil	disp	hp	drat	wt	qsec	vs
Mazda RX4	21,0	6	160,0	110	3,90	2,620	16,46	0
Mazda RX4 Wag	21,0	6	160,0	110	3,90	2,875	17,02	0
Datsun 710	22,8	4	108,0	93	3,85	2,320	18,61	1
Hornet 4 Drive	21,4	6	258,0	110	3,08	3,215	19,44	1
Hornet Sportcarga de 18,7	8	360,0	175	3,15	3,440	17,02	0	
Valente	18,1	6	225,0	105	2,76	3,460	20,22	1
Espanador 360	14,3	8	360,0	245	3,21	3,570	15,84	0
Merc 240D	24,4	4	146,7	62	3,69	3,190	20,00	1
Merc 230	22,8	4	140,8	95	3,92	3,150	22,90	1
Merc 280	19,2	6	167,6	123	3,92	3,440	18,30	1

**TABELA 2.3:** Um Conto de Duas Tabelas.

Sepal.Length	Sepal.Width	mpg	cil	disp
5.1	3,5	Mazda RX4	21,0	6 160
4.9	3,0	Mazda RX4 Wag	21,0	6 160
4.7	3,2	Datsun 710	22,8	4 108
		Hornet 4 Drive	21,4	6 258
		Hornet Sportcarga de 18,7	8	360

um exemplo. Observe que esse recurso está disponível apenas em HTML e Saída PDF.

```
tricotar::kable(
 Lista(
 cabeça(íris[, 1:2], 3),
 head(mtcars[, 1:3], 5)
),
 caption = 'Um Conto de Duas Mesas.', booktabs = TRUE
)
```

Quando você não quer que uma tabela flutue em PDF, você pode usar o LaTeX

pacote **longtable**,<sup>7</sup> que pode quebrar uma tabela em várias páginas. Para usar **longtable**, passe `longtable = TRUE` para `kable()`, e certifique-se de incluir `\usepackage{longtable}` no preâmbulo do LaTeX (veja Seção 4.1 para como personalizar o preâmbulo do LaTeX). Claro, isso é irrelevante à saída HTML, pois as tabelas em HTML não precisam flutuar.

```
tricotar::kable(
 iris[1:55,], longtable = TRUE, booktabs = TRUE,
 caption = 'Uma tabela gerada pelo pacote longtable.'
)
```

**TABELA 2.4:** Uma tabela gerada pelo pacote `longtable`.

Sepal.Comprimento	Sepal.Largura	Pétala.Comprimento	Pétala.Largura	Espécie
5.1	3,5	1,4	0,2	sedoso
4.9	3,0	1,4	0,2	sedoso
4.7	3,2	1,3	0,2	sedoso
4.6	3,1	1,5	0,2	sedoso
5,0	3,6	1,4	0,2	sedoso
5.4	3,9	1,7	0,4	sedoso
4,6	3,4	1,4	0,3	sedoso
5,0	3,4	1,5	0,2	sedoso
4,4	2,9	1,4	0,2	sedoso
4.9	3,1	1,5	0,1	sedoso
5.4	3,7	1,5	0,2	sedoso
4,8	3,4	1,6	0,2	sedoso
4,8	3,0	1,4	0,1	sedoso
4,3	3,0	1,1	0,1	sedoso
5,8	4,0	1,2	0,2	sedoso
5.7	4,4	1,5	0,4	sedoso
5.4	3,9	1,3	0,4	sedoso
5.1	3,5	1,4	0,3	sedoso
5.7	3,8	1,7	0,3	sedoso

<sup>7</sup> <https://www.ctan.org/pkg/longtable>

5,1	3,8	1,5	0,3 sedoso
5,4	3,4	1,7	0,2 sedoso
5,1	3,7	1,5	0,4 sedoso
4,6	3,6	1,0	0,2 sedoso
5,1	3,3	1,7	0,5 sedoso
4,8	3,4	1,9	0,2 sedoso
5,0	3,0	1,6	0,2 sedoso
5,0	3,4	1,6	0,4 sedoso
5,2	3,5	1,5	0,2 sedoso
5,2	3,4	1,4	0,2 sedoso
4,7	3,2	1,6	0,2 sedoso
4,8	3,1	1,6	0,2 sedoso
5,4	3,4	1,5	0,4 sedoso
5,2	4,1	1,5	0,1 sedoso
5,5	4,2	1,4	0,2 sedoso
4,9	3,1	1,5	0,2 sedoso
5,0	3,2	1,2	0,2 sedoso
5,5	3,5	1,3	0,2 sedoso
4,9	3,6	1,4	0,1 sedoso
4,4	3,0	1,3	0,2 sedoso
5,1	3,4	1,5	0,2 sedoso
5,0	3,5	1,3	0,3 sedoso
4,5	2,3	1,3	0,3 sedoso
4,4	3,2	1,3	0,2 sedoso
5,0	3,5	1,6	0,6 sedoso
5,1	3,8	1,9	0,4 sedoso
4,8	3,0	1,4	0,3 sedoso
5,1	3,8	1,6	0,2 sedoso
4,6	3,2	1,4	0,2 sedoso
5,3	3,7	1,5	0,2 sedoso
5,0	3,3	1,4	0,2 sedoso
7,0	3,2	4,7	1,4 versicolor
6,4	3,2	4,5	1,5 versicolor
6,9	3,1	4,9	1,5 versicolor

5,5	2,3	4,0	1,3 versicolor
6,5	2,8	4,6	1,5 versicolor

O Pandoc suporta vários tipos de tabelas Markdown,<sup>8</sup> como tabelas simples, tabelas multilinhas, tabelas de grade e tabelas de pipe. o que knitr::kable() gera é uma tabela simples como esta:

**Tabela:** Uma tabela simples no Markdown.

Sepal.Comprimento Sepal.Width Petal.Length Petal.Width

5,1	3,5	1,4	0,2
4,9	3,0	1,4	0,2
4,7	3,2	1,3	0,2
4,6	3,1	1,5	0,2
5,0	3,6	1,4	0,2
5,4	3,9	1,7	0,4

Você pode usar qualquer tipo de tabela Markdown em seu documento. Ser capaz de fazer referência cruzada a uma tabela Markdown, ela deve ter uma legenda rotulada no formato Tabela: (#label) Legenda aqui, onde rótulo deve ter o prefixo tab:, por exemplo, tab: tabela simples.

Se você decidir usar outros pacotes R para gerar tabelas, terá para garantir que o rótulo do ambiente da tabela apareça no início da legenda da tabela no formato (\#label) (novamente, o rótulo deve tem a aba prefixo :). Você tem que ter muito cuidado com a *portabilidade* da função de geração de tabela: deve funcionar tanto para HTML quanto para Saída LaTeX automaticamente, por isso deve considerar o formato de saída internamente(verifique knitr::opts\_knit\$get('rmarkdown.pandoc.to')).Quando escrevendo uma tabela HTML, a legenda deve ser escrita na tag <caption></caption> . Para tabelas simples, kable() deve ser suficiente. Se você ter que criar tabelas complicadas (por exemplo, com certas células abrangendo em várias colunas/linhas), você terá que levar em consideração as questões acima mencionadas.

<sup>8</sup> <http://pandoc.org/MANUAL.html#tables>

## 2.6 Referências cruzadas

Explicitamos como as referências cruzadas funcionam para equações (Seção 2.2.1), teoremas (Seção 2.2.2), figuras (Seção 2.4) e tabelas (Seção 2.5). Na verdade, você também pode fazer referência a seções usando o mesmo syn tax \@ref(label), onde label é o ID da seção. Por padrão, Pandoc irá gerar um ID para todos os cabeçalhos de seção, por exemplo, uma seção # Hello World irá tem um ID hello-world. Recomendamos que você atribua manualmente um ID a um cabeçalho de seção para garantir que você não esqueça de atualizar o rótulo de referência depois de alterar o cabeçalho de seção. Para atribuir um ID a um cabeçalho da seção, basta adicionar {#id} ao final do cabeçalho da seção. Outros atributos de cabeçalhos de seção podem ser definidos usando o Pandoc padrão sintaxe<sup>9</sup>.

Quando um rótulo referenciado não pode ser encontrado, você verá duas perguntas marcas como ??, bem como uma mensagem de aviso no console R ao renderizar o livro.

Você também pode criar links baseados em texto usando a seção explícita ou automática IDs ou até mesmo o texto do cabeçalho da seção real.

- Se você estiver satisfeito com o cabeçalho da seção como o texto do link, use-o dentro um único conjunto de colchetes:
  - [Texto do cabeçalho da seção]: exemplo “Um único documento” via [Um único documento]
- Há duas maneiras de especificar o texto do link personalizado:
  - [texto do link][texto do cabeçalho da seção], por exemplo, “livros não ingleses” via [livros não ingleses][Internacionalização]
  - [texto do link](#ID), por exemplo, “coisas de mesa” via [coisas de mesa](#tabelas)

A documentação do Pandoc fornece mais detalhes sobre IDs de seção automáticas<sup>10</sup> e referências de cabeçalho implícitas.<sup>11</sup>

As referências cruzadas ainda funcionam mesmo quando nos referimos a um item que não está

<sup>9</sup> <http://pandoc.org/MANUAL.html#heading-identifiers>

<sup>10</sup> [http://pandoc.org/MANUAL.html#extension-auto\\_identifiers](http://pandoc.org/MANUAL.html#extension-auto_identifiers)

<sup>11</sup> [http://pandoc.org/MANUAL.html#extension-implicit\\_header\\_references](http://pandoc.org/MANUAL.html#extension-implicit_header_references)

a página atual da saída PDF ou HTML. Por exemplo, veja a Equação (2.1) e a Figura 2.4.

---

## 2.7 Blocos personalizados

Blocos personalizados são frequentemente usados em livros técnicos para criar caixas salientes de código e/ou narrativa que chamem a atenção do leitor. Por exemplo, blocos personalizados podem ser usados para destacar uma nota ou um aviso. Estes podem ser incluídos em vários formatos de saída de **bookdown** usando o syn tax do Pandoc para blocos Div protegidos (<https://pandoc.org/MANUAL.html#divs-and-spans>). Seção 9.6 no *R Markdown Cookbook*<sup>12</sup> (Xie et al., 2020) para instruções.

O formato de saída HTML bs4\_book() inclui estilo para blocos personalizados selecionados; consulte a Seção 3.1.2.

---

## 2.8 Citações

Pandoc oferece dois métodos para gerenciar citações e referências bibliográficas. referências em um documento.

1. O método padrão é usar um programa auxiliar Pandoc chamado pandoc-citeproc<sup>13</sup>, que segue as especificações da Citation Style Language (CSL)<sup>14</sup> e obtém instruções específicas de formatação de um dos inúmeros

Arquivos de estilo CSL.<sup>15</sup>

2. Os usuários também podem optar por usar **natbib**<sup>16</sup> (baseado em bibtex) ou **biblatex**<sup>17</sup> como um "pacote de citações". Neste caso, a bibliografia

<sup>12</sup><https://bookdown.org/yihui/rmarkdown-cookbook/custom-blocks.html>

<sup>13</sup><https://github.com/jgm/pandoc-citeproc>

<sup>14</sup><https://docs.citationstyles.org/en/v1.0.1/specification.html>

<sup>15</sup><https://www.zotero.org/styles/>

<sup>16</sup><https://ctan.org/pkg/natbib>

<sup>17</sup><https://ctan.org/pkg/biblatex>

arquivos de dados gráficos precisam estar no formato bibtex ou biblatex , e o formato de saída do documento é limitado a PDF. Novamente, vários estilos bibliográficos estão disponíveis (favor consultar o documento desses pacotes).

Para usar **natbib** ou **biblatex** para processar referências, você pode definir a opção citation\_package da saída R Markdown para mat, por exemplo,

```
resultado:
pdf_document:
 Citation_package: natbib
bookdown::pdf_book:
 Citation_package: biblatex
```

Mesmo se você escolher natbib ou biblatex para saída em PDF, todos os outros formatos de saída usarão pandoc-citeproc. Se você usar estilos correspondentes exemplo, biblio-style: apa citeproc, a saída PDF é: para o tipo de saída PDF. Por será muito semelhante, embora não necessariamente idênticas.

Para qualquer formato de saída não PDF, pandoc-citeproc é o único opção. Se a consistência entre os formatos de saída PDF e não PDF for importante, use pandoc-citeproc por toda parte.

Os dados bibliográficos podem estar em vários formatos. Nós apenas mostramos exemplos de bancos de dados BibTeX nesta seção e consulte a seção “Citações”<sup>18</sup> do manual Pandoc para outros formatos possíveis.

Um banco de dados BibTeX é um arquivo de texto simples (com o nome de arquivo convencional extensão .bib) que consiste em entradas de bibliografia como esta:

```
@Manual{R-base,
 title = {R: Um idioma e ambiente para estatística
 Informática},
 autor = {{R Core Team}}},
```

---

<sup>18</sup><https://pandoc.org/MANUAL.html#citations>

```

organização = {R Foundation for Statistical Computing}, endereço = {Vienna,
Austria},
ano = {2016}, url =
{https://www.R-project.org/},
}

```

Uma entrada de bibliografia começa com @type{}, onde tipo pode ser artigo, livro, manual e assim por diante.<sup>19</sup> Em seguida, há uma chave de citação, como R-base no exemplo acima. Para citar uma entrada, use @key ou [@key] (o último coloca a citação entre chaves), por exemplo, @R-base é renderizado como **R Core Team (2022)**, e [@R-base] gera “(R Equipe Principal, 2022)”. Uma nota pode ser incluída entre colchetes, por exemplo, [uma nota sobre, @R-base] será renderizada como “(uma nota sobre, **R Core Team, 2022**)”. Se você estiver familiarizado com o pacote **natbib** no LaTeX, @key é basicamente \citet{key}, e [@key] é equivalente a \citep{key}.

Há vários campos em uma entrada bibliográfica, como título, autor e ano, etc. Você pode ver <https://en.wikipedia.org/wiki/BibTeX> para possíveis tipos de entradas e campos no BibTeX.

Existe uma função auxiliar `write_bib()` no **knitr** para gerar entradas BibTeX automaticamente para pacotes R, por exemplo,

```

o segundo argumento pode ser um arquivo .bib
knitr::write_bib(c("knitr", "stringr"), "", largura = 60)

```

```

@Manual{R-knitr, title
= {knitr: A General-Purpose Package for Dynamic
Geração de Relatórios em R},
autor = {Yihui Xie},
ano = {2022}, nota
= {R pacote versão 1.39},
url = {https://yihui.org/knitr/},
}

```

---

<sup>19</sup>O nome do tipo não diferencia maiúsculas de minúsculas, portanto, não importa se é manual, Manual ou MANUAL.

```

@Manual{R-stringr,
 title = {stringr: Wrappers simples e consistentes para comuns
 Operações de String},
 autor = {Hadley Wickham},
 ano = {2019}, nota
 = {R pacote versão 1.4.0},
 url = {https://CRAN.R-project.org/package=stringr},
}

@Livro{knitr2015,
 title = {Documentos Dinâmicos com {R} e knitr},
 autor = {Yihui Xie}, editor =
 {Chapman e Hall/CRC},
 endereço = {Boca Raton, Flórida}, ano =
 {2015},
 edição = {2ª}, nota =
 {ISBN 978-1498716963},
 url = {https://yihui.org/knitr/},
}

@InCollection{knitr2014,
 booktitle = {Implementando Computação Reprodutível
 Pesquisar},
 editor = {Victoria Stodden e Friedrich Leisch e Roger
 D. Peng},
 title = {knitr: uma ferramenta abrangente para reprodução
 Pesquisa em {R}}, autor
 = {Yihui Xie},
 editora = {Chapman e Hall/CRC},
 ano = {2014},
 nota = {ISBN 978-1466561595},
 url = {http://www.crcpress.com/product/isbn/
 9781466561595},
}

```

Depois de ter um ou vários arquivos .bib , você pode usar a bibliografia de campo nos metadados YAML do seu primeiro documento R Markdown

## 2.9 Índice

43

(que normalmente é index.Rmd), e você também pode especificar o estilo de bibliografia via biblio-style (isso se aplica apenas à saída PDF), por exemplo,

```
...
bibliografia: ["one.bib", "another.bib", "yet-another.bib"] biblio-style: "apalike"
```

```
citações de links: true
...
```

O campo link-citations pode ser usado para adicionar links internos do texto de citação do estilo autor-ano à entrada de bibliografia na saída HTML.

Quando o formato de saída for LaTeX, a lista de referências será automaticamente colocada em um capítulo ou seção no final do documento. Para saída não LaTeX, você pode adicionar um capítulo vazio como o último capítulo do seu livro. Por exemplo, se seu último capítulo for o arquivo Rmd 06-references.Rmd, seu conteúdo pode ser uma expressão R inline:

```
`r if (knitr::is_html_output()) '# Referências {-}'
```

Para instruções mais detalhadas e mais exemplos sobre como usar citações, consulte a seção “Citações” do manual Pandoc.

---

## 2.9 Índice

Atualmente, o índice é suportado apenas para saída LaTeX/PDF. Para imprimir um índice após o livro, você pode usar o pacote LaTeX **makeidx** no preâmbulo (consulte a Seção 4.1):

```
\usepackage{makeidx}
\makeindex
```

Em seguida, insira \printindex no final do seu livro através da opção YAML

ção inclui -> after\_body. Uma entrada de índice pode ser criada por meio do comando \index{} no corpo do livro, por exemplo, \index{GIT}.

## 2.10 widgets HTML

Embora um dos maiores pontos fortes do R seja a visualização de dados, existem um grande número de bibliotecas JavaScript para uma visualização de dados muito mais rica. Essas bibliotecas podem ser usadas para construir aplicativos interativos que pode renderizar facilmente em navegadores da Web, para que os usuários não precisem instalar nenhum pacotes de software adicionais para visualizar as visualizações. Uma maneira de trazer essas bibliotecas JavaScript para o R é através do **htmlwidgets**<sup>20</sup> pacote (Vaidyanathan et al., 2021).

Widgets HTML podem ser renderizados como uma página da Web independente (como um R plot) ou incorporado em documentos R Markdown e aplicativos Shiny.

Eles foram originalmente projetados apenas para saída HTML e exigem a disponibilidade de JavaScript, portanto, eles não funcionarão em formatos de saída não HTML, como LaTeX/PDF. Antes **do knitr** v1.13, você obteria um erro ao renderizar widgets HTML para um formato de saída que não é HTML. Desde **o knitr** v1.13, os widgets HTML serão renderizados automaticamente como capturas de tela feitas através do pacote **webshot** (Chang, 2022). Do Claro, você precisa instalar o pacote **webshot**. Além disso, você tem que instalar o PhantomJS (<http://phantomjs.org>), já que é o que o **web shot** usa para capturar capturas de tela. Tanto **o webshot** quanto o PhantomJS podem ser instalado automaticamente a partir do R:

```
install.packages("webshot")
webshot::install_phantomjs()
```

A função `install_phantomjs()` funciona para Windows, OS X e Linux. Você também pode optar por baixar e instalar o PhantomJS por você mesmo, se estiver familiarizado com a modificação do ambiente do sistema variável PATH.

---

<sup>20</sup><http://htmlwidgets.org>

## 2.10 widgets HTML

Quando o **knitr** detecta um objeto widget HTML em um pedaço de código, ele renderiza o widget normalmente quando o formato de saída atual é HTML, ou salva o widget como uma página HTML e chama o **webshot** para capturar o tela da página HTML quando o formato de saída não for HTML. Aqui é um exemplo de uma tabela criada a partir do pacote DT ([Xie et al., 2022](#)):

DT::datatable(íris)

Show 10 entries					Search:
	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5	3.6	1.4	0.2	setosa
6	5.4	3.9	1.7	0.4	setosa
7	4.6	3.4	1.4	0.3	setosa
8	5	3.4	1.5	0.2	setosa
9	4.4	2.9	1.4	0.2	setosa
10	4.9	3.1	1.5	0.1	setosa

Showing 1 to 10 of 150 entries

Previous 1 2 3 4 5 ... 15 Next

**FIGURA 2.5:** Um widget de tabela renderizado por meio do pacote DT.

Se você estiver lendo este livro como páginas da web agora, deverá ver uma tabela interativa gerada a partir do trecho de código acima, por exemplo, você pode classifique as colunas e pesquise na tabela. Se você estiver lendo uma versão não HTML deste livro, deverá ver uma captura de tela da tabela.

A captura de tela pode parecer um pouco diferente com o widget real renderizado no navegador da Web, devido à diferença entre uma tela real navegador e navegador virtual do PhantomJS.

Existem várias opções de **knitr** chunk relacionadas à captura de tela. Primeiro, se você não estiver satisfeito com a qualidade das capturas de tela automáticas ou quiser uma captura de tela do widget de um determinado estado (por exemplo, depois de clicar e ordenar uma determinada coluna de uma tabela), você pode capturar a tela manualmente e fornecer sua própria captura de tela

através da opção de bloco screenshot.alt (capturas de tela alternativas). Esta opção toma os caminhos das imagens. Se você tiver vários widgets em um bloco, você pode fornecer um vetor de caminhos de imagem. Quando esta opção estiver presente, **O knitr** não chamará mais o **webshot** para fazer capturas de tela automáticas.

Segundo, às vezes você pode querer forçar o **knitr** a usar capturas de tela estáticas em vez de renderizar os widgets reais mesmo em páginas HTML. Dentro neste caso, você pode definir a opção chunk screenshot.force = TRUE, e widgets sempre serão renderizados como imagens estáticas. Observe que você ainda pode optar por usar capturas de tela automáticas ou personalizadas.

Terceiro, o **webshot** tem algumas opções para controlar as capturas de tela automáticas, e você pode especificar essas opções através da opção chunk screen shot.opts, que recebe uma lista como list(delay = 2, cliprect = 'view port'). Consulte a página de ajuda ? webshot::webshot21 para obter a lista completa de possíveis opções, e a introdução do pacote22 para uma ilustração do efeito de algumas dessas opções. Aqui a opção de atraso pode ser importante para widgets que demoram muito para renderizar: delay especifica o número segundos para esperar antes que o PhantomJS faça a captura de tela. Se você ver uma captura de tela incompleta, você pode querer especificar um atraso maior (o padrão é 0,2 segundos).

Quarto, se você acha que é lento para capturar as capturas de tela ou não quer para fazer isso toda vez que o pedaço de código é executado, você pode usar o pedaço opção cache = TRUE para armazenar em cache o pedaço. O cache funciona tanto para HTML e formatos de saída não HTML.

As capturas de tela se comportam como gráficos R normais no sentido de que muitos opções de blocos relacionadas a figuras também se aplicam a capturas de tela, incluindo fig.width, fig.height, out.width, fig.cap e assim por diante. Então você pode especificar o tamanho das capturas de tela no documento de saída e atribua legendas de figuras a elas também. O formato de imagem das capturas de tela automáticas pode ser especificado através da opção chunk dev, e os valores possíveis são pdf, png, e jpeg. O padrão para saída PDF é pdf, e é png para outros tipos de saída. Observe que o pdf pode não funcionar tão fielmente quanto o png: algumas vezes há certos elementos em uma página HTML que não são renderizados para a captura de tela do PDF, então você pode querer usar dev = 'png' mesmo para PDF

---

21`https://wch.github.io/webshot/reference/webshot.html`

22`https://wch.github.io/webshot/articles/intro.html`

## 2.11 Páginas da Web e aplicativos Shiny

resultado. Depende de casos específicos de widgets HTML, e você pode tentar pdf e png (ou jpeg ) antes de decidir qual formato é mais desejável.

### 2.11 Páginas da Web e aplicativos Shiny

Semelhante aos widgets HTML, páginas da Web arbitrárias podem ser incorporadas no livro. Você pode usar a função knitr::include\_url() para incluir um web página por meio de seu URL. Quando o formato de saída é HTML, um iframe é usado;<sup>23</sup> em outros casos, o knitr tenta fazer uma captura de tela da web página (ou use a captura de tela personalizada que você forneceu). Todas as opções de pedaços são os mesmos dos widgets HTML. Uma opção que pode exigir sua atenção especial é a opção de atraso : widgets HTML são renderizados localmente, então geralmente eles são rápidos para carregar para o PhantomJS fazer capturas de tela, mas um URL arbitrário pode demorar mais para carregar, então você pode querer para usar um valor de atraso maior , por exemplo, use a opção chunk screenshot.opts = lista(atraso = 5).

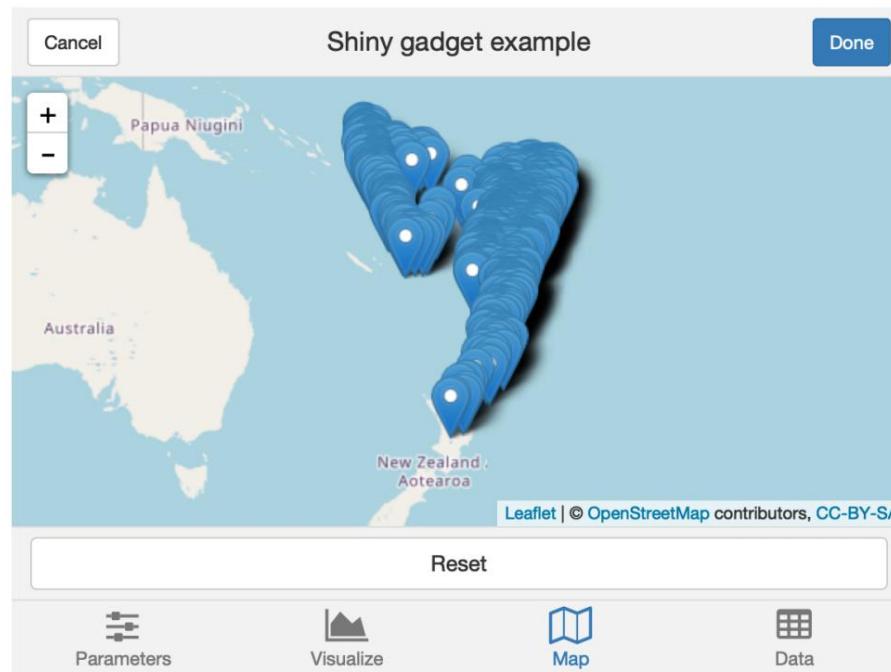
Uma função relacionada é knitr::include\_app(), que é muito semelhante a include\_url(), e foi projetada para incorporar aplicativos Shiny por meio de seus URLs na saída. Sua única diferença com include\_url() é que ele adiciona automaticamente um parâmetro de consulta ?showcase=0 ao URL, se nenhum outro parâmetros de consulta estão presentes na URL, para desabilitar o modo de exibição brilhante, que provavelmente não será útil para capturas de tela ou iframes. Se você quiser o modo de apresentação, use include\_url() em vez de include\_app().

Abaixo está um exemplo de aplicativo Shiny (Figura 2.6):

```
knitr::include_app("https://yihui.shinyapps.io/minUI/",
altura = "600px")
```

Novamente, você verá um aplicativo ao vivo se estiver lendo uma versão HTML do este livro e uma captura de tela estática se você estiver lendo outros tipos de formatos. O aplicativo Shiny acima foi criado usando o pacote **miniUI** (Cheng, 2018), que fornece funções de layout que são particularmente

<sup>23</sup>Um iframe é basicamente uma caixa em uma página da web para incorporar outra página da web.



**FIGURA 2.6:** Um aplicativo Shiny criado por meio do pacote miniUI; você pode ver uma versão ao vivo em <https://yihui.shinyapps.io/miniUI/>.

bom para aplicativos Shiny em telas pequenas. Se você usar as funções normais de layout Shiny, provavelmente verá barras de rolagem verticais e/ou horizontais nos iframes porque o tamanho da página é grande demais para caber em um iframe. Quando a largura padrão do iframe é muito pequena, você pode usar a opção chunk out.width para alterá-la. Para a altura do iframe, use o argumento height de include\_url() / include\_app().

Aplicativos brilhantes podem demorar ainda mais para carregar do que URLs comuns. Você pode querer usar um valor conservador para a opção de atraso , por exemplo, 10. Desnecessário dizer que include\_url() e include\_app() requerem uma conexão de Internet funcional, a menos que você tenha armazenado em cache o fragmento (mas as páginas da Web dentro de iframes ainda irão não funcionar sem uma conexão com a Internet).

# 3

## *Formatos de saída*

O pacote **bookdown** suporta principalmente três tipos de saída para mats: HTML, LaTeX/PDF e e-books. Neste capítulo, apresentamos as opções possíveis para esses formatos. Os formatos de saída podem ser especificados nos metadados YAML do primeiro arquivo Rmd do livro ou em um arquivo YAML separado chamado `_output.yml` no diretório raiz do livro. Aqui está um breve exemplo do primeiro (os formatos de saída são especificados no campo de saída dos metadados YAML):

```
--
 título: "Um livro impressionante"
 autor: "Li Lei e Han Meimei"
 saída:
 bookdown::gitbook:
 lib_dir: ativos
 split_by: configuração
 da seção :
 barra de ferramentas:
 posição: estática
 bookdown::pdf_book:
 keep_tex: true
 bookdown::html_book:
 css: toc.css
 documentclass: livro
--
```

Aqui está um exemplo de `_output.yml`:

```
bookdown::gitbook:
 lib_dir: assets
```

```

split_by: configuração da
seção :
barra de ferramentas:
posição: bookdown
estático ::pdf_book:
keep_tex: verdadeiro
bookdown::html_book:
css: toc.css

```

Nesse caso, todos os formatos devem estar no nível superior, em vez de em um campo de saída . Você não precisa dos três traços --- em \_output.yml.

### 3.1 HTML

A principal diferença entre renderizar um livro (usando **bookdown**) com renderizar um único documento R Markdown (usando **rmarkdown**) para HTML é que um livro irá gerar várias páginas HTML por padrão—normalmente um arquivo HTML por capítulo. Isso torna mais fácil marcar um determinado capítulo ou compartilhar seu URL com outras pessoas enquanto você lê o livro e mais rápido carregar um livro no navegador da web. Atualmente, fornecemos vários estilos diferentes para saída HTML:

- o estilo GitBook (Seção 3.1.1), • o
- estilo Bootstrap de três colunas (Seção 3.1.2), • o
- estilo Bootstrap padrão (Seção 3.1.3) e • o estilo Tufte (Seção 3.1.4).

#### 3.1.1 Estilo GitBook

O estilo GitBook foi emprestado do GitBook, um projeto lançado pela Friendcode, Inc. (<https://www.gitbook.com>) e dedicado a ajudar autores a escrever livros com Markdown. Ele fornece um estilo bonito, com um layout que consiste em uma barra lateral mostrando o índice à esquerda e o corpo principal de um livro à direita. O design é responsivo ao tamanho da janela, por exemplo, os botões de navegação são exibidos

a esquerda/direita do corpo do livro quando a janela é larga o suficiente e recolhida na parte inferior quando a janela é estreita para dar aos leitores mais espaço horizontal para ler o corpo do livro.

A maneira mais fácil de começar a escrever um novo gitbook é usar o RStudio Project Wizard (*File > New Project > New Directory > Book project using bookdown*) e selecionar gitbook no menu suspenso (veja a Figura 3.3).

Se você não usa RStudio ou prefere uma função, você pode criar o mesmo template de projeto com `bookdown::create_gitbook()` do seu console R. Veja `?bookdown::create_gitbook` para ajuda.

Fizemos várias melhorias em relação ao projeto GitBook original.

O mais significativo é que substituímos o mecanismo Markdown pelo R Markdown v2 baseado no Pandoc, para que haja muito mais recursos para você usar ao escrever um livro:

- Você pode incorporar trechos de código R e expressões R inline em Markdown, e isso facilita a criação de documentos reproduzíveis e libera você de sincronizar sua computação com sua saída real (`knitr` cuidará disso automaticamente). • A sintaxe do Markdown é muito mais rica: você pode escrever qualquer coisa que o Markdown do Pandoc suporte, como expressões matemáticas LaTeX e citações.
- Você pode incorporar conteúdo interativo no livro (somente para saída HTML), como widgets HTML e aplicativos Shiny.

Também adicionamos alguns recursos úteis na interface do usuário que apresentaremos em detalhes em breve. A função de formato de saída para o estilo Git Book no `bookdown` é `gitbook()`. Seguem seus argumentos:

```
gitbook(fig_caption = TRUE, number_sections = TRUE, self_contained
= FALSE, anchor_sections = TRUE,
lib_dir = "libs", global_numbering = !number_sections, pandoc_args =
NULL, ..., template = "default",
split_by = c("capítulo", "capítulo+número", "seção",
"seção+número", "rmd", "nenhum"), split_bib = TRUE,
config = list(), table_css = TRUE)
```

A maioria dos argumentos são passados para `rmarkdown::html_document()`, incluindo `fig_caption`, `lib_dir` e .... Você pode conferir a página de ajuda do `rmarkdown::html_document()` para obter a lista completa de opções possíveis. Nós fortemente recomendo que você use `fig_caption = TRUE` por dois motivos: 1) é importante explicar suas figuras com legendas; 2) habilitar legendas de figuras significa que as figuras serão colocadas em ambientes flutuantes quando o saída é LaTeX, caso contrário você pode acabar com muito espaço em branco no determinadas páginas. O formato dos números das figuras/tabelas depende se as seções são numeradas ou não: se `number_sections = TRUE`, esses números terá o formato  $Xi$ , onde  $X$  é o número do capítulo, e  $i$  em um número incremental; se as seções não forem numeradas, todas as figuras/tabelas serão numerados sequencialmente ao longo do livro de 1, 2, ...,  $N$ . Nota que em ambos os casos, as figuras e tabelas serão numeradas independentemente.

Entre todos os argumentos possíveis em ..., é mais provável que você use o `css` argumento para fornecer um ou mais arquivos CSS personalizados para ajustar o padrão Estilo CSS. Existem alguns argumentos de `html_document()` que foram gitbook() embutido no código e você não pode alterá-los: `toc = TRUE` (há deve ser um índice), `theme = NULL` (não usando nenhum Bootstrap temas) e `template` (existe um template interno do GitBook).

Observe que se você alterar `self_contained = TRUE` para criar páginas HTML independentes, o tamanho total de todos os arquivos HTML pode ser significativamente aumentado, pois há muitos arquivos JS e CSS que precisam ser embutido em cada arquivo HTML.

Além dessas opções `html_document()` , `gitbook()` tem três outros argumentos: `split_by`, `split_bib` e `config`. O argumento `split_by` especifica como você deseja dividir a saída HTML em várias páginas e seus valores possíveis são:

- `rmd`: use os nomes de arquivo base dos arquivos Rmd de entrada para criar o HTML nomes de arquivos, por exemplo, gere `capítulo3.html` para `capítulo3.Rmd`.
- `nenhum`: não divida o arquivo HTML (o livro será um único arquivo HTML).
- `capítulo`: divide o arquivo pelos cabeçalhos de primeiro nível.
- `seção`: divide o arquivo pelos cabeçalhos de segundo nível.
- `capítulo+número` e `seção+número`: semelhante ao `capítulo` e `seção`, mas os arquivos serão numerados.

Para `capítulo` e `seção`, os nomes dos arquivos HTML serão determinados por

os identificadores de cabeçalho, por exemplo, o nome do arquivo para o primeiro capítulo com um título de capítulo # Introdução será por padrão como introdução.html . Para capítulo+número e seção+número, os números do capítulo/seção serão anexados aos nomes dos arquivos HTML, por exemplo, 1-introduction.html e 2-1-literature.html. O identificador do cabeçalho é gerado automaticamente a partir do texto do cabeçalho por padrão,<sup>1</sup> e você pode especificar manualmente um identificador usando a sintaxe {#your-custom-id} após o texto do cabeçalho, por exemplo,

```
Uma introdução {#introduction}
```

O identificador padrão é `an-introduction` mas nós o alteramos para `introduction`.

Por padrão, a bibliografia é dividida e os itens de citação relevantes são colocados na parte inferior de cada página, para que os leitores não precisem navegar para uma página de bibliografia diferente para ver os detalhes das citações. Este recurso pode ser desabilitado usando split\_bib = FALSE, neste caso todas as citações são colocadas em uma página separada.

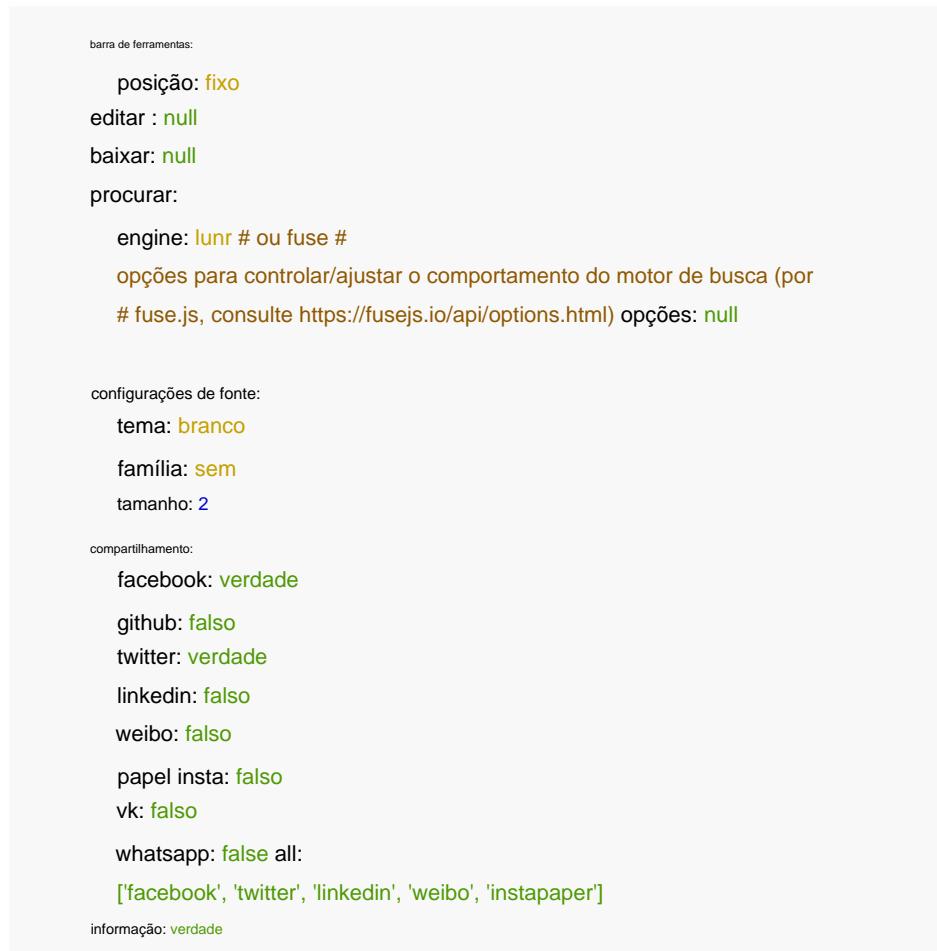
Existem várias subopções na opção de configuração para você ajustar alguns detalhes na interface do usuário. Lembre-se de que todas as opções de formato de saída (não apenas para bookdown::gitbook) podem ser passadas para a função de formato se você usar a interface de linha de comando bookdown::render\_book() ou gravadas nos metadados YAML. Exibimos as subopções padrão de configuração no formato gitbook como metadados YAML abaixo (observe que elas são recuadas na opção de configuração):

```
bookdown::gitbook: config:

 toc:
 recolher: subseção scroll_highlight:
 true
 antes: null
 depois: null
```

---

<sup>1</sup>Para ver mais detalhes sobre como um identificador é gerado automaticamente, veja a extensão auto\_identifiers na documentação do Pandoc <http://pandoc.org/MANUAL.html#header-identifiers>



A opção sumário controla o comportamento do sumário (TOC). Você pode recolher alguns itens inicialmente quando uma página é carregada por meio da opção de recolher . Seus valores possíveis são subseção, seção, nenhum (ou nulo). Essa opção pode ser útil se seu sumário for muito longo e tiver mais de três níveis de títulos: subseção significa recolher todos os itens do sumário para subseções (XXX), seção significa que os itens para seções (XX) exibido inicialmente, e nenhum significa não recolher nenhum item no sumário. Para esses itens de sumário recolhidos, você pode alternar sua visibilidade clicando nos itens de sumário pai. Por exemplo, você pode clicar no título de um capítulo no sumário para mostrar/ocultar suas seções.

A opção scroll\_highlight em toc indica se deve-se habilitar high-

iluminação dos itens do sumário à medida que você rola o corpo do livro (por padrão, esse recurso está ativado). Sempre que um novo cabeçalho entrar na porta de visualização atual enquanto você rola para baixo/para cima, o item correspondente no sumário à esquerda será realçado.

Como a barra lateral tem uma largura fixa, quando um item no sumário é truncado porque o texto do cabeçalho é muito largo, você pode passar o cursor sobre ele para ver uma dica de ferramenta mostrando o texto completo.

Você pode adicionar mais itens antes e depois do sumário usando a tag HTML `<li>`. Esses itens serão separados do TOC usando um divisor horizontal. Você pode usar o caractere pipe | para que você não precise escapar nenhum caractere nesses itens seguindo a sintaxe YAML, por exemplo,

`toc:`

```
antes: |
Meu livro incrível
João Smith
depois: |
 Orgulhosamente
publicado com bookdown
```

Conforme você navega pelas diferentes páginas HTML, tentaremos preservar a posição de rolagem do sumário. Normalmente, você verá a barra de rolagem no sumário em uma posição fixa, mesmo que navegue para a próxima página. No entanto, se o item de sumário do capítulo/seção atual não estiver visível quando a página for carregada, rolaremos automaticamente o sumário para torná-lo visível para você.



**FIGURA 3.1:** A barra de ferramentas do GitBook.

O estilo GitBook tem uma barra de ferramentas (Figura 3.1) no topo de cada página que

permite alterar dinamicamente as configurações do livro. A opção da barra de ferramentas possui uma posição de subopção , que pode assumir valores fixos ou estáticos. O padrão é que a barra de ferramentas será fixada na parte superior da página, portanto, mesmo que você role a página para baixo, a barra de ferramentas ainda estará visível lá. Se for estático, a barra de ferramentas não rolará com a página, ou seja, quando você rolar para fora, não a verá mais.

O primeiro botão na barra de ferramentas pode alternar a visibilidade da barra lateral. Você também pode pressionar a tecla S no teclado para fazer a mesma coisa. O estilo GitBook pode lembrar o status de visibilidade da barra lateral, por exemplo, se você fechou a barra lateral, ela permanecerá fechada na próxima vez que você abrir o livro. Na verdade, o estilo do GitBook também lembra muitas outras configurações, como a palavra-chave de pesquisa e as configurações de fonte.

O segundo botão na barra de ferramentas é o botão de pesquisa. Seu atalho de teclado é F (Localizar). Quando o botão for clicado, você verá uma caixa de pesquisa na parte superior da barra lateral. Conforme você digita na caixa, o sumário será filtrado para exibir as seções que correspondem à palavra-chave de pesquisa. Agora você pode usar as teclas de seta para cima/para baixo para destacar a correspondência anterior/seguinte nos resultados da pesquisa. Quando você clicar no botão de pesquisa novamente (ou pressionar F fora da caixa de pesquisa), a palavra-chave de pesquisa será esvaziada e a caixa de pesquisa ficará oculta. Para desabilitar a pesquisa, defina a opção search: false no config.

O terceiro botão é para configurações de fonte/tema. O leitor pode alterar o tamanho da fonte (maior ou menor), a família da fonte (serif ou sans serif) e o tema (branco, sépia ou noturno). Você pode definir o valor inicial dessas configurações por meio da opção fontsettings . O tamanho da fonte é medido em uma escala de 0 a 4; o valor inicial pode ser definido como 1, 2 (padrão), 3 ou 4. O botão pode ser removido da barra de ferramentas configurando fontsettings: null (ou no).

### # alterando o padrão

configurações de fonte:

tema: noite

família: serif

tamanho: 3

A opção de edição é a mesma que a mencionada na Seção 4.4. Se não estiver vazio, um botão de edição será adicionado à barra de ferramentas. Isso foi de-

assinado para que potenciais colaboradores do livro contribuam editando o livro no GitHub depois de clicar no botão e enviar solicitações de pull.

As opções de histórico e visualização funcionam da mesma maneira.

Se o seu livro tiver outros formatos de saída para download pelos leitores, você pode fornecer a opção de download para que um botão de download possa ser adicionado à barra de ferramentas. Esta opção aceita um vetor de caractere ou uma lista de vetores de caractere com o comprimento de cada vetor sendo 2. Quando for um vetor de caractere, deverá ser um vetor de nomes de arquivo ou extensões de nome de arquivo, por exemplo, ambos os seguintes as configurações estão ok:

```
download: ["book.pdf", "book.epub"] download:
["pdf", "epub", "mobi"]
```

Quando você fornece apenas as extensões de nome de arquivo, o nome de arquivo é derivado do nome de arquivo do livro do arquivo de configuração `_bookdown.yml` (Seção 4.4). Quando o download for nulo, `gitbook()` procurará arquivos PDF, EPUB e MOBI no diretório de saída do livro e os adicionará automaticamente à opção de download. Se você deseja apenas suprimir o botão de download, use `download: false`. Todos os arquivos para download dos leitores serão exibidos em um menu suspenso e as extensões de nome de arquivo serão usadas como texto do menu.

Quando o único formato disponível para download for PDF, o botão de download será um único botão PDF em vez de um menu suspenso.

Uma forma alternativa para o valor da opção de download é uma lista de vetores de comprimento 2, por exemplo,

```
download: [["book.pdf", "PDF"], ["book.epub", "EPUB"]]
```

Você também pode escrevê-lo como:

```
download:
- ["livro.pdf", "PDF"]
- ["livro.epub", "EPUB"]
```

Cada vetor na lista consiste no nome do arquivo e no texto a ser exibido.

jogado no menu. Comparado com o primeiro formulário, este formulário permite que você para personalizar o texto do menu, por exemplo, você pode ter duas cópias diferentes de o PDF para os leitores baixarem e você precisará fazer o menu itens diferentes.

À direita da barra de ferramentas, existem alguns botões para compartilhar o link sites de redes sociais como Twitter, Facebook e Linkedin. Você pode usar a opção de compartilhamento para decidir quais botões habilitar. Se você quiser se livrar totalmente desses botões, use o compartilhamento: nulo (ou não).

Outro botão mostrado na barra de ferramentas é o botão information('i') que lista os atalhos de teclado disponíveis para navegar no documento. Este botão pode ser ocultado definindo info: false.

Por fim, há mais algumas opções de nível superior nos metadados YAML que pode ser passado para o modelo HTML do GitBook via Pandoc. Elas podem não ter efeitos visíveis claros na saída HTML, mas podem ser útil quando você implanta a saída HTML como um site. Essas opções incluir:

- descrição: uma cadeia de caracteres a ser gravada no atributo content da tag <meta name="description" content=""> no cabeçalho HTML (se faltar, será usado o título do livro). Isso pode ser útil para otimização de mecanismos de busca (SEO). Observe que deve ser texto simples sem qualquer formatação Markdown, como \_italic\_ ou \*\*negrito\*\*.
- URL: o URL por exemplo, do livro local na rede Internet,  
2  
<https://bookdown.org/yihui/bookdown/>.
- github-repo: O repositório GitHub do livro do formulário user/repo.
- imagem da capa: O caminho para a imagem da capa do livro.
- ícone de toque de maçã: um caminho para um ícone (por exemplo, uma imagem PNG). Isso é para Somente iOS: quando o site é adicionado à tela inicial, o link é representado por este ícone.
- apple-touch-icon-size: o tamanho do ícone (por padrão, 152 x 152 pix eles).
- favicon: Um caminho para o “ícone favorito”. Normalmente este ícone é exibido

---

<sup>2</sup>A barra invertida antes de : é devido a um problema técnico: queremos evitar Pandoc de traduzir o link para código HTML <a href="..."></a>. Mais detalhes em <https://github.com/jgm/pandoc/issues/2139>.

na barra de endereços do navegador ou na frente do título da página na guia se o navegador suportar guias.

Abaixo, mostramos alguns metadados YAML de amostra (mas uma vez, observe que essas são opções de nível superior):

```
--
título: "Um livro incrível"
autor: "João Smith"
description: "Este livro apresenta a teoria ABC e ..." url: 'https://bookdown.org/
john/awesome/'
github-repo: "john/awesome" imagem
de capa: "images/cover.png"
apple-touch-icon: "touch-icon.png" apple-touch-
icon-size: 120
favicon: "favicon.ico"
--
```

Um bom efeito de definir a descrição e a imagem da capa é que quando você compartilha o link do seu livro em alguns sites de redes sociais como o Twitter, o link pode ser automaticamente expandido para um cartão com a imagem da capa e a descrição do livro.

### 3.1.2 Estilo Bootstrap de três colunas

O formato de saída bs4\_book() é construído com Bootstrap (<https://getbootstrap.com>), usando recursos cuidadosamente criados para fornecer uma experiência de leitura limpa, esteja você em um telefone, tablet ou desktop. Em uma tela de tamanho normal, o layout inclui três colunas de conteúdo para que os leitores possam ver rapidamente todos os capítulos à esquerda, o capítulo atual no meio e as seções dentro do capítulo atual à direita.

Você pode ler um livro de exemplo aqui: <https://mastering-shiny.org> Além dos componentes básicos de **bookdown** (Seção 2), os principais recursos do bs4\_book são:

- Fácil personalização de cores e fontes com o pacote **bslib**.<sup>4</sup>

---

<sup>4</sup> <https://pkgs.rstudio.com/bslib/>

The screenshot shows the 'Welcome' page of the 'Mastering Shiny' book. The left sidebar contains a search bar and a table of contents with chapters from 'Getting started' to 'Mastering reactivity'. The main content area features a large image of a green bird (Hadley Wickham) next to the book cover, which includes the title 'Mastering Shiny' and the subtitle 'Build Interactive Apps, Reports & Dashboards Powered by R'. The right sidebar includes links for 'On this page' (Welcome, License), 'View source' (with a GitHub icon), and 'Edit this page' (with a GitHub icon).

3

**FIGURA 3.2:** Página inicial de um livro no estilo Bootstrap de três colunas.

- Pesquisa integrada (dividida por seção) que ajuda os leitores a encontrar rapidamente o que estão procurando.
- Uma barra lateral contendo um índice dentro do capítulo que facilita a navegação e ajuda a fornecer contexto sobre sua posição atual no capítulo.
- Tipografia pensada para tornar o conteúdo o mais fácil de ler possível, independentemente do tamanho do seu dispositivo. Um cabeçalho adesivo sai do seu caminho durante a leitura, mas é facilmente acessível se você precisar.
- Notas de rodapé em linha significam que você pode ler apartes ao lado do texto a que se referem. Este tema combina melhor com um estilo de referência que gera notas de rodapé.
- O realce da sintaxe R e a vinculação automática pelo pacote `downlit`<sup>5</sup> são combinados com um esquema de cores acessível projetado por Alison Hill.
- Metadados aprimorados para compartilhamento social por meio de plataformas como Twitter,

5 <https://downlit.r-lib.org>

LinkedIn e Facebook, para que cada capítulo compartilhado tenha uma descrição única, gerada automaticamente com base no conteúdo desse capítulo.

- Capacidade de configurar links para um repositório remoto como GitHub ou Git Lab, permitindo que os leitores visualizem facilmente o arquivo fonte de cada capítulo ou sugira uma edição.

A função de formato de saída é `bookdown::bs4_book`.

<sup>6</sup> Aqui estão seus argumentos

```
bs4_book(theme = bs4_book_theme(), repo = NULL, ...,
 lib_dir = "libs", pandoc_args = NULL, extra_dependencies = NULL, template = "default", split_bib = FALSE,
 footnotes_inline = TRUE)
```

### 3.1.2.1 Escrevendo um `bs4_book`

A maneira mais fácil de começar a escrever um novo `bs4_book` é usar o RStudio Project Wizard (*File > New Project > New Directory > Book project using bookdown*) e selecionar `bs4_book` no menu suspenso (veja a Figura 3.3).

Se você não usa RStudio ou prefere uma função, você pode criar o mesmo template de projeto com `bookdown::create_bs4_book()` do seu console R. Consulte `?bookdown::create_bs4_book` ou a documentação online<sup>7</sup> para obter ajuda.

Este estilo é projetado para livros que usam um capítulo por página. Isso significa que cada capítulo é um arquivo `.Rmd`, e cada arquivo `.Rmd` pode conter um capítulo. Cada arquivo *deve* começar com um título de primeiro nível, # título do capítulo, e esse deve ser o único título de primeiro nível no arquivo.

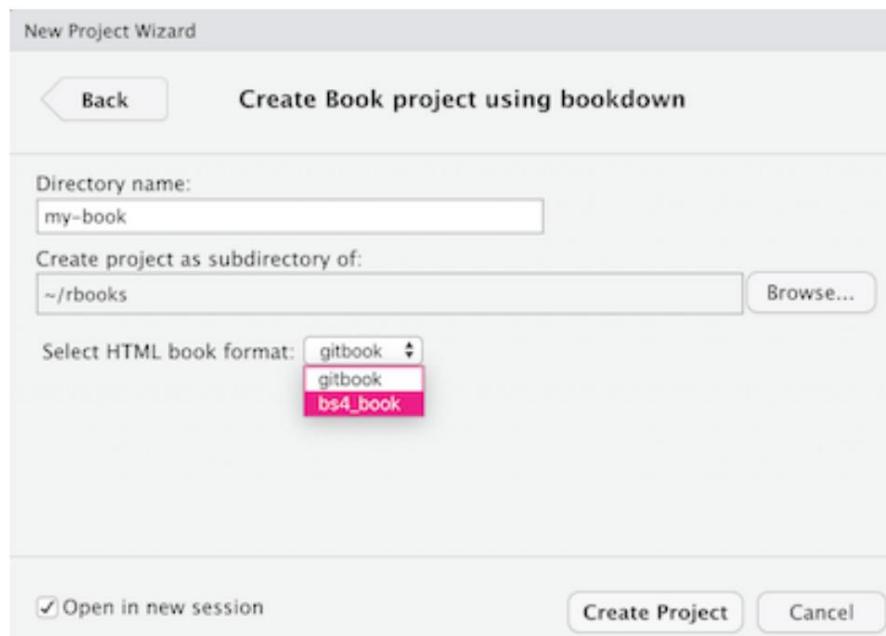
Use títulos de segundo nível e de nível inferior em capítulos como:

```
Um capítulo
Uma secção
```

---

<sup>6</sup> [https://pkgs.rstudio.com/bookdown/reference/bs4\\_book.html](https://pkgs.rstudio.com/bookdown/reference/bs4_book.html)

<sup>7</sup> [https://pkgs.rstudio.com/bookdown/reference/create\\_book.html](https://pkgs.rstudio.com/bookdown/reference/create_book.html)



**FIGURA 3.3:** Captura de tela do RStudio Project Wizard para criar um novo projeto bookdown.

```
Uma subseção
```

Os títulos de primeiro e segundo níveis aparecem na barra lateral do capítulo atual, que fica no topo da página à medida que você rola para baixo. Quando uma seção é navegada, os subtítulos de terceiro nível, como "Uma subseção", são expandidos automaticamente.

O arquivo index.Rmd é necessário e também é seu primeiro capítulo de livro. Será a página inicial quando você renderizar o livro. Se você quiser incluir conteúdo que só deve ser incluído na versão HTML do livro, você pode querer incluir esse conteúdo condicionalmente combinando a opção `knitr` `include` chunk com a função `knitr::is_html_output()`. Consulte o *R Markdown Cookbook*<sup>8</sup> para obter instruções.

Um cabeçalho YAML em index.Rmd para um bs4\_book ficaria assim:

---

<sup>8</sup> <https://bookdown.org/yihui/rmarkdown-cookbook/latex-html.html>

```

title: "Um exemplo mínimo de livro"
autor: "Jane Doe"
data: "2022-07-28"
site: bookdown::bookdown_site
saída: bookdown::bs4_book url:
https://bookdown.org/janedoe/bookdown-demo
imagem da capa: cover.png
descrição: |
 Este é um exemplo mínimo de uso do pacote bookdown para escrever um livro.
 O formato de saída para este exemplo é bookdown::bs4_book.

```

### 3.1.2.2 Estilo e personalização

O formato `bs4_book()` baseia-se na estrutura CSS Bootstrap (versão 49), uma biblioteca de código aberto de partes reutilizáveis de código HTML, CSS e JavaScript. O framework Bootstrap permite fácil customização de cores e fontes através do pacote **bslib** R.

Você pode usar a opção de tema para adicionar uma cor primária em hexadecimal para mat,10 que mudará a cor de todos os links em seu livro e a cor de fundo do rodapé.

```
bookdown::bs4_book:
 tema:
 primário: "#0d6efd"
```

Para configurações de fonte personalizadas, adicionar uma palavra-chave `google:` aciona a capacidade do `sass::font_google()`'s 11 de importar automaticamente arquivos do Google Font.12 Aqui está um exemplo de YAML que altera `base_font`, `header_font` e `code_font`:

---

9 <https://getbootstrap.com/docs/4.0/>  
 10 [https://en.wikipedia.org/wiki/Web\\_colors](https://en.wikipedia.org/wiki/Web_colors)  
 11 [https://rstudio.github.io/sass/reference/font\\_face.html](https://rstudio.github.io/sass/reference/font_face.html)  
 12 <https://fonts.google.com>

```
bookdown::bs4_book:
 tema:
 primário: "#0d6efd"
 fonte_base:
 google: isso
 Fonte Cabeçalho:
 o Google:
 família: amargo
 peso: 200
 code_font:
 google:
 # argumentos para sass::font_google()
 família: DM Mono
 local: falso
```

Por padrão, google: agrupará arquivos de fonte com seu livro, para que ele carregue, armazene em cache e exiba os arquivos de fonte relevantes localmente. Isso significa que quando você compartilha com outra pessoa, as fontes são garantidas para renderização, mesmo sem uma conexão com a Internet (local: false importa arquivos via URL em vez de servi-los localmente).

Você também pode usar fontes que não são do Google veiculadas localmente usando 13 sass::font\_face().

#### 3.1.2.3 Blocos de texto explicativo

Os blocos de texto explicativo podem ser usados para destacar certas partes do conteúdo do restante da narrativa. O estilo bs4\_book inclui blocos de texto explicativo especiais com estilos predefinidos para adicionar uma borda colorida ao redor do texto e/ou código dentro do texto explicativo. Use o seguinte syn tax para criar um bloco de texto explicativo:

```
::: {.rmdnote}
O estilo `bs4_book` também inclui um bloco de texto explicativo `rmdnote`
como este.
```

<sup>13</sup>[https://rstudio.github.io/sass/reference/font\\_face.html#serving-non-google fonts-locally](https://rstudio.github.io/sass/reference/font_face.html#serving-non-google fonts-locally)

```
```{r colapso=TRUE}
head(beaver1, n = 5)
```

⋮

Você pode usar a sintaxe Markdown e o código embutido dentro de um bloco. Quando tricotado, a saída será semelhante à Figura 3.4.

The `bs4_book` theme also includes an `.rmdnote` callout block like this one.

```
head(beaver1, n = 5)
#>   day time temp activ
#> 1 346 840 36.33    0
#> 2 346 850 36.34    0
#> 3 346 900 36.35    0
#> 4 346 910 36.42    0
#> 5 346 920 36.55    0
```

FIGURA 3.4: Um bloco de texto explicativo especial.

Os blocos disponíveis são: `.rmdnote`, `.rmdcaution`, `.rmdimportant`, `.rmdtip` e `.rmdwarning`. As cores usadas serão baseadas nas cores padrão fornecidas pelo Bootstrap, mas também podem ser personalizadas em seu arquivo `_output.yml`:

```
bookdown::bs4_book:
  tema:
    primário: "#0d6efd" # default .rmdnote = perigo azul: "#dc3545"
    # default .rmdcaution = vermelho
    sucesso: "#198754" # default .rmdimportant = green
    informações: "#0dcaf0" # default .rmdtip = ciano
    aviso: "#ffc107" # default .rmdwarning = amarelo
```

Para saída LaTeX, apenas o conteúdo desses blocos será mostrado com

nenhum contorno colorido como para HTML. Cabe ao usuário definir a aparência desses blocos para saída LaTeX usando ambientes personalizados. Consulte o *R Markdown Cookbook*¹⁴ para obter instruções.

3.1.2.4 Metadados HTML

Bookdown irá gerar tags HTML <meta> com base nas variáveis do Pandoc definidas em index.Rmd, descritas em [6.4.2](#). Além disso, bs4_book() criará descrições de capítulos únicas geradas automaticamente a partir do conteúdo do capítulo. Você pode dar uma olhada no template HTML bs4_book¹⁵ para detalhes sobre como essas variáveis são usadas.

3.1.2.5 Notas de Rodapé Inline

bs4_book faz qualquer nota de rodapé para mostrar em linha ao passar o mouse em vez de um item vinculado na parte inferior da página. Você pode definir footnotes_inline = FALSE para desativar esse comportamento e manter as notas de rodapé na parte inferior.

```
bookdown::bs4_book:  
  footnotes_inline: false
```

3.1.2.6 Referências/Bibliografia

Fazer suas citações de notas de rodapé permite que os leitores as leiam perto do texto onde são usadas porque bs4_book faz com que as notas de rodapé apareçam em linha quando clicadas. Para fazer isso, baixe um arquivo CSL estilo nota de rodapé; recomendamos <https://www.zotero.org/styles/>. Por exemplo, você pode baixar o chicago-fullnote-bibliography.csl do Zotero,¹⁶ e adicioná-lo ao seu index.Rmd:

```
bibliografia: refs.bib  
csl: chicago-fullnote-bibliography.csl
```

Opcionalmente, se você não quiser mais uma seção de referência no final do livro, adicione esta linha ao seu index.Rmd:

¹⁴<https://bookdown.org/yihui/rmarkdown-cookbook/custom-blocks.html>

¹⁵https://github.com/rstudio/bookdown/blob/main/inst/templates-bs4_book.html

¹⁶<https://www.zotero.org/styles/?q=id%3Achicago-fullnote-bibliography>

3.1 HTML

67

```
suprimir-bibliografia: true
```

Se você quiser usar um estilo de citação que não suporte notas de rodapé, as referências não serão mostradas em linha nos pop-ups. Nesse caso, você pode querer adicionar a opção `split_bib` ao seu `_output.yml`:

```
bookdown::bs4_book:  
  split_bib: verdadeiro
```

Em seguida, sua bibliografia será dividida e os itens de citação relevantes serão colocados na parte inferior de cada capítulo, para que os leitores não precisem navegar para uma página de bibliografia diferente para ver os detalhes das citações.

3.1.2.7 Especificando o repositório

Especifique um repositório de origem para o seu livro para dar aos leitores a opção de visualizar facilmente o arquivo de origem de cada capítulo ou sugerir uma edição.

Se o seu livro tiver uma ramificação padrão chamada “principal”, você poderá usar:

```
bookdown::bs4_book:  
  repositório:  
    base: https://github.com/hadley/ggplot2-book  
    ramo: principal
```

Se o seu livro estiver localizado em um subdiretório chamado “book”, você pode usar:

```
bookdown::bs4_book:  
  repositório:  
    base: https://github.com/hadley/ggplot2-book  
    ramo: principal  
    subdiretório: livro
```

Por padrão, se o URL do repositório contiver “github”, ele receberá um ícone GitHub Font Awesome¹⁷, caso contrário, obterá um ícone GitLab. Para usar outro ícone,

¹⁷<https://fontawesome.com>

especifique-o com o prefixo correto, como fas, fab e assim por diante (Font Awe some 518).

```
bookdown::bs4_book:

repositorio:
  base: https://github.com/hadley/ggplot2-book
  ramo: principal
  subdiretorio: livro
  ícone: "fas fa-air-freshner"
```

3.1.3 O estilo padrão do Bootstrap

Se você já usou R Markdown antes, você deve estar familiarizado com o Estilo Bootstrap (<https://getbootstrap.com>), qual é o estilo padrão da saída HTML do R Markdown. A função de formato de saída em **rmarkdown** é `html_document()`, e temos um correspondente para `mat html_book()` em **bookdown** usando `html_document()` como base para `mat`. Você pode ler um exemplo `html_book()` aqui: <https://bookdown.org/yihui/bookdown-demo2>

Na verdade, existe um formato mais geral `html_chapters()` no **bookdown** e `html_book()` é apenas seu caso especial:

```
html_chapters(toc = TRUE, number_sections = TRUE, fig_caption = TRUE,
  lib_dir = "libs", template = bookdown_file("templates/default.html"),
  global_numbering = !number_sections, pandoc_args = NULL,
  ..., base_format = rmarkdown::html_document, split_bib = TRUE,
  page_builder = build_chapter, split_by = c("seção+número",
  "seção", "capítulo+número", "capítulo", "rmd", "nenhum"))
```

Observe que ele tem um argumento `base_format` que recebe uma saída base para a função `mat`, e `html_book()` é basicamente `html_chapters(base_format = rmarkdown::html_document)`. Todos os argumentos de `html_book()` são passados para `html_chapters()`:

¹⁸<https://fontawesome.com/v5.0/how-to-use/on-the-web/referencing-icons/basic-usar>

```
html_book(...)
```

Isso significa que você pode usar a maioria dos argumentos de rmark down::html_document, como toc (para mostrar o índice), number_sections (para numerar títulos de seção) e assim por diante. Novamente, verifique a página de ajuda do rmarkdown::html_document para ver a lista completa de opções possíveis. Observe que o argumento self_contained é codificado internamente para FALSE , portanto, você não pode alterar o valor desse argumento. Explicamos o argumento split_by na seção anterior.

Os argumentos template e page_builder são para usuários avançados, e você não precisa entendê-los a menos que você tenha uma forte necessidade de customizar a saída HTML, e essas muitas opções fornecidas por rmarkdown::html_document() ainda não fornecem o que você deseja.

Se você deseja passar um modelo HTML diferente para o argumento do modelo , o modelo deve conter três pares de comentários HTML e cada comentário deve estar em uma linha separada:

- <!--bookdown:title:start--> e <!--bookdown:title:end--> para marcar a seção de título do livro. Esta seção será colocada apenas na primeira página do livro renderizado;
 - <!--bookdown:toc:start--> e <!--bookdown:toc:end--> para marcar a seção de sumário, que será colocada em todas as páginas HTML;
 - <!--bookdown:body:start--> e <!--bookdown:body:end--> para marcar o corpo HTML do livro, e o corpo HTML será dividido em várias páginas separadas.
- Lembre-se de que mesclamos todos os arquivos R Markdown ou Markdown, os renderizamos em um único arquivo HTML e o dividimos.

Você pode abrir o modelo HTML padrão para ver onde esses comentários foram inseridos:

```
bookdown:::bookdown_file("templates/default.html") # você pode
usar file.edit() para abrir este arquivo
```

Depois de saber como o **bookdown** funciona internamente para gerar saída HTML de várias páginas, será mais fácil entender o argumento

`page_builder`, que é uma função para compor cada página HTML individual usando os fragmentos HTML extraídos dos tokens de comentário acima. O valor padrão de `page_builder` é uma função `build_chapter` em `bookdown`, e seu código fonte é relativamente simples (ignore as funções internas como `button_link()`):

```
build_chapter = function(
  head, toc, capítulo, link_prev, link_next, rmd_cur, html_cur, foot ) { toc = add_toc_class(toc)

  colar(c(
    cabeça,
    '<div class="row">', '<div',
    class="col-sm-12">',
    bater,
    '</div>',
    '</div>',
    '<div class="row">',
    '<div class="col-sm-12">',
    capítulo,
    '<p style="text-align: center;">',
    button_link(link_prev, 'Anterior'),
    source_link(rmd_cur, type = 'edit'),
    source_link(rmd_cur, type = 'history'),
    source_link(rmd_cur, type = 'view'),
    button_link(link_next, 'Próximo'), '</p>',

    '</div>',
    '</div>',
    pé
  ), recolher = '\n')
}
```

Basicamente, essa função recebe vários componentes, como o cabeçalho HTML, o índice, o corpo do capítulo e assim por diante, e espera-se que retorne uma cadeia de caracteres que é a fonte HTML de um componente.

página HTML completa. Você pode manipular todos os componentes desta função usando funções de processamento de texto como `gsub()` e `paste()`.

O que o construtor de páginas padrão faz é colocar o sumário na primeira linha, o corpo na segunda linha, botões de navegação na parte inferior do corpo e concatená-los com o cabeçalho e o rodapé do HTML. Aqui está um esboço do código-fonte HTML que pode ajudá-lo a entender a saída de `build_chapter()`:

```
<html>
  <cabeça>
    <title>Um bom livro</title>
  </head>
  <corpo>

    <div class="row">TOC</div>

    <div class="linha">
      CORPO DO CAPÍTULO
      <p>
        <button>ANTERIOR</button>
        <button>PRÓXIMO</button>
      </p>
    </div>

  </body>
</html>
```

Para todas as páginas HTML, a principal diferença é o corpo do capítulo, e a maioria dos demais elementos são os mesmos. A saída padrão de `html_book()` incluirá os arquivos Bootstrap CSS e JavaScript na tag `<head>`.

O TOC é frequentemente usado para fins de navegação. No estilo GitBook, o sumário é exibido na barra lateral. Para o estilo Bootstrap, não aplicamos um estilo especial a ele, então ele é mostrado como uma lista simples e não ordenada (na tag HTML ``). É fácil transformar essa lista em uma barra de navegação com algumas técnicas de CSS. Fornecemos um arquivo CSS `toc.css` neste

pacote que você pode usar, e você pode encontrá-lo aqui: <https://github.com/rstudio/bookdown/blob/master/inst/examples/css/toc.css>

Você pode copiar este arquivo para o diretório raiz do seu livro e aplicá-lo à saída HTML por meio da opção `css`, por exemplo,

```
--  
resultado:  
bookdown::html_book:  
  toc: verdade  
  css: toc.css  
--
```

Há muitas maneiras possíveis de transformar listas `` em menus de navegação se você pesquisar um pouco na web, e você pode escolher um estilo de menu que você gosta. O `toc.css` que acabamos de mencionar é um estilo com textos de menu brancos em um fundo preto e suporta submenus (por exemplo, títulos de seções são exibidos como menus suspensos sob títulos de capítulos).

Na verdade, você pode se livrar do estilo Bootstrap em `html_document()` se você definir a opção `theme` como `null`, e você está livre para aplicar estilos arbitrários à saída HTML usando a opção `css` (e possivelmente a opção `includes` se você deseja incluir conteúdo arbitrário no cabeçalho/pé do HTML).

3.1.4 Estilo tufado

Assim como o estilo Bootstrap, o estilo Tufte é fornecido por uma saída para `tufte_html_book()`, que também é um caso especial de `html_chapters()` usando `tufte::tufte_html()` como formato base. Por favor, veja a idade do pacote **tufte** (Xie e Allaire, 2022) se você não estiver familiarizado com o estilo Tufte.

Você pode ler um exemplo `tufte_html_book()` aqui: <https://bookdown.org/yihui/bookdown-demo3/>

Basicamente, é um layout com uma coluna principal à esquerda e uma coluna de margem à direita. O corpo principal está na coluna principal e a coluna de margem é usada para colocar notas de rodapé, notas de margem, referências e figuras de margem e assim por diante.

Todos os argumentos de `tufte_html_book()` têm exatamente os mesmos significados

como `html_book()`, por exemplo, você também pode personalizar o CSS através da opção `css`. No entanto, existem alguns elementos específicos do estilo Tufte, como notas de margem, figuras de margem e figuras de largura total. Esses elementos requerem sintaxe especial para serem gerados; consulte a documentação do pacote de **tufos**. Observe que você não precisa fazer nada especial com notas de rodapé e referências (basta usar a sintaxe normal de Markdown `^[[nota de rodapé]` e `[@citação]`), pois elas serão colocadas automaticamente na margem. Um breve exemplo YAML do formato `tufte_html_book`:

```
-->
resultado:
bookdown::tufte_html_book:
  toc: verdade
  css: toc.css
-->
```

3.2 LaTeX/PDF

Recomendamos fortemente que você use um formato de saída HTML em vez de LaTeX ao desenvolver um livro, pois você não ficará muito distraído com os detalhes de composição, o que pode incomodá-lo muito se você olhar constantemente para a saída PDF de um livro. Deixe o trabalho de composição cuidadosa para o final (de preferência depois de ter realmente terminado o conteúdo do livro).

O formato de saída LaTeX/PDF é fornecido por `pdf_book()` em **book down**. Não há uma diferença significativa entre `pdf_book()` e o formato `pdf_document()` no **rmarkdown**. O objetivo principal de `pdf_book()` é resolver os rótulos e referências cruzadas escritas usando a sintaxe descrita nas Seções 2.4, 2.5 e 2.6. Se o único formato de saída que você deseja para um livro é LaTeX/PDF, você pode usar a sintaxe específica do LaTeX, como `\label{}` para rotular figuras/tabelas/seções e `\ref{}` para fazer referência cruzada a elas via seus rótulos, porque o Pandoc suporta comandos LaTeX no Markdown. No entanto, a sintaxe do LaTeX não é portátil para outros formatos de saída, como HTML e e-books. É por isso que nós

introduziu a sintaxe (`\#label`) para rótulos e `\@ref(label)` para referências cruzadas.

Existem algumas opções YAML de nível superior que serão aplicadas à saída do La TeX. Para um livro, você pode alterar a classe de documento padrão para livro (o padrão é artigo) e especificar um estilo de bibliografia exigido pelo seu editor. Um breve exemplo de YAML:

```
---
documentclass: livro
bibliografia: [book.bib, packages.bib]
estilo bibliográfico: apalike
---
```

Há um grande número de outras opções YAML que você pode especificar para saída LaTeX, como tamanho do papel, tamanho da fonte, margem da página, espaçamento entre linhas, famílias de fontes e assim por diante. Consulte <http://pandoc.org/MANUAL.html#variables-for-latex> para obter uma lista completa de opções.

O formato `pdf_book()` é um formato geral como `html_book()`, e também possui um argumento `base_format`:

```
pdf_book(toc = TRUE, number_sections = TRUE, fig_caption = TRUE,
         pandoc_args = NULL, ..., base_format = rmarkdown::pdf_document,
         toc_unnumbered = TRUE, toc_appendix = FALSE, toc_bib = FALSE, quote_footer
         = NULL, highlight_bw = FALSE)
```

Você pode alterar a função `base_format` para outra saída para funções `mat`, e `bookdown` forneceu uma função wrapper simples `tufte_book2()`, que é basicamente `pdf_book(base_format = tufte::tufte_book)`, para produzir um livro PDF usando o estilo Tufte PDF (novamente , veja o pacote de `tufos`).

3.3 E-books

Atualmente **bookdown** oferece dois formatos de e-book, EPUB e MOBI.

Livros nesses formatos podem ser lidos em dispositivos como smartphones, tablets ou leitores eletrônicos especiais, como o Kindle.

3.3.1 EPUB

Para criar um livro EPUB, você pode usar o formato `epub_book()`. Tem algumas opções em comum com `rmarkdown::html_document()`:

```
epub_book(fig_width = 5, fig_height = 4, dev = "png",
           fig_caption = TRUE, number_sections = TRUE, toc = FALSE,
           toc_depth = 3, stylesheet = NULL, cover_image = NULL, metadados =
           NULL, capítulo_level = 1, epub_version = c("epub3",
           "epub", "epub2"), md_extensions = NULL, global_numbering = !number_sections,
           pandoc_args = NULL, template = "padrão")
```

A opção sumário está desativada porque o leitor de e-book muitas vezes pode descobrir um sumário automaticamente a partir do livro, portanto, não é necessário adicionar algumas páginas para o sumário. Existem algumas opções específicas para EPUB:

- folha de estilo: É semelhante à opção `css` nos formatos de saída HTML e você pode personalizar a aparência dos elementos usando CSS.
- `cover_image`: O caminho para a imagem da capa do livro.
- `metadados`: O caminho para um arquivo XML para os metadados do livro (consulte documentação do Pandoc para mais detalhes).
- `nível_capítulo`: Internamente, um livro EPUB é uma série de arquivos “capítulo”, e esta opção determina o nível pelo qual o livro é dividido nesses arquivos. Isso é semelhante ao argumento `split_by` dos formatos de saída HTML que mencionamos na Seção 3.1, mas um livro EPUB é um arquivo único e você não verá esses arquivos de “capítulo” diretamente. O nível padrão é o primeiro nível, e se você definir como 2, significa que o livro será organizado internamente por arquivos de seção, o que pode permitir que o leitor carregue o livro mais rapidamente.
- `epub_version`: Versão 3 ou 2 do EPUB.

Um livro EPUB é essencialmente uma coleção de páginas HTML, por exemplo, você pode aplicar regras CSS a seus elementos, incorporar imagens, inserir expressões matemáticas (porque o MathML é parcialmente suportado) e assim por diante. Figura/tabela legendas, referências cruzadas, blocos personalizados e citações mencionadas em O Capítulo 2 também funciona para EPUB. Você pode comparar a saída EPUB de este livro para a saída HTML, e você verá que a única diferença é a aparência visual.

Existem vários leitores EPUB disponíveis, incluindo Calibre (<https://www.calibre-ebook.com>), iBooks da Apple e Google Play Livros.

3.3.2 MOBI

Os e-books MOBI podem ser lidos nos dispositivos Kindle da Amazon. Pandoc faz não suporta saída MOBI nativamente, mas você pode usar ferramentas de terceiros para converter EPUB para MOBI. Uma ferramenta possível é o Calibre. O Calibre é de código aberto e gratuito, e suporta conversão entre muitos outros formatos.

Por exemplo, você pode converter HTML para EPUB, documentos do Word para MOBI, e assim por diante. A função `calibre()` no **bookdown** é um wrapper função do utilitário de linha de comando `ebook-convert` no Calibre. Você precisa ter certeza de que o `ebook-convert` executável pode ser encontrado via a variável de ambiente PATH. Se você usa o macOS, pode instalar o Calibre com o Homebrew (<https://brew.sh>) através do comando `brew cask install calibre`, então você não precisa se preocupar com o problema do PATH .

3.4 Um único documento

Às vezes, você pode não querer escrever um livro, mas um único artigo ou relatório longo. Normalmente o que você faz é chamar `rmark down::render()` com um certo formato de saída. As principais funcionalidades que faltam são a numeração automática de figuras/tabelas/equações, e referências cruzadas de figuras/tabelas/equações/seções. Nós fatoramos esses recursos do **bookdown**, para que você possa usá-los sem ter que preparar um livro de vários arquivos Rmd.

As funções `html_document2()`, `tufte_html2()`, `pdf_document2()`,

3.4 Um único documento

`word_document2()`, `tufte_handout2()` e `tufte_book2()` são projetados para esta finalidade. Se você renderizar um documento R Markdown com o formato de saída, digamos, `bookdown::html_document2`, você obterá números de figuras/tabelas e poderá fazer referência cruzada a eles em uma única página HTML usando a sintaxe descrita no Capítulo 2.

Abaixo estão alguns exemplos desses formatos de saída nos metadados YAML de um único arquivo Rmd (você também pode adicionar esses formatos ao arquivo `_out_put.yml`):

```
saída:
  bookdown::html_document2: default
  bookdown::pdf_document2:
    keep_tex: true
  bookdown::word_document2:
    toc: verdade
```

As funções de formato de saída HTML e PDF acima são basicamente wrappers de formatos de saída `bookdown::html_book` e `bookdown::pdf_book`, no sentido de que eles mudaram o argumento `base_format`. Por exemplo, você pode dar uma olhada no código-fonte de `pdf_document2`:

```
bookdown::pdf_document2

## função ...
## {
##   pdf_book(..., base_format = rmarkdown::pdf_document)
## }
## <bytecode: 0x7f82e2f6e400>
## <ambiente: namespace:bookdown>
```

Depois de conhecer esse fato, você pode aplicar a mesma ideia a outros formatos de saída usando o formato_base apropriado. Por exemplo, você pode portar os recursos de **bookdown** para o formato `jss_article` no pacote **rticles** (Allaire et al., 2022a) usando os metadados YAML:

resultado:

```
bookdown::pdf_book:  
base_format: rticles::jss_article
```

Então você poderá usar todos os recursos que apresentamos no Capítulo 2.

Embora o formato gitbook() tenha sido projetado principalmente para livros, você também pode aplicá-lo a um único documento R Markdown. A única diferença é que não haverá botão de pesquisa na saída de uma única página, porque você pode simplesmente usar a ferramenta de pesquisa do seu navegador da Web para encontrar o texto (por exemplo, pressione Ctrl + F ou Command + F). Você também pode definir a opção split_by como none para gerar apenas uma única página de saída, nesse caso não haverá botões de navegação, pois não há outras páginas para navegar. Você ainda pode gerar arquivos HTML de várias páginas, se desejar. Outra opção que você pode querer usar é self_contained = TRUE quando é apenas uma única página de saída.

4

Costumização

Como mencionamos no início deste livro, espera-se que você tenha algum conhecimento básico sobre R Markdown, e estamos nos concentrando em introduzir os recursos de **bookdown** em vez de **rmark down**. Na verdade, o R Markdown é altamente personalizável e há muitas opções que você pode usar para personalizar o documento de saída. Dependendo de quanto você deseja personalizar a saída, você pode usar algumas opções simples nos metadados YAML ou apenas substituir todo o modelo Pandoc.

4.1 Opções YAML

Para a maioria dos tipos de formatos de saída, você pode personalizar os estilos de realce de sintaxe usando a opção de realce do formato específico. Atualmente, os estilos possíveis são default, tango, pygments, kate, monochrome, espresso, zenburn, haddock e brisadark. Por exemplo, você pode escolher o estilo de tango para o formato gitbook :

```
--  
resultado:  
  bookdown::gitbook:  
    destaque: tango  
--
```

Para formatos de saída HTML, é mais provável que você use a opção css para fornecer suas próprias folhas de estilo CSS para personalizar a aparência dos elementos HTML. Existe uma opção inclui que se aplica a mais mats, incluindo HTML e LaTeX. A opção inclui permite que você

insira conteúdo personalizado arbitrário antes e/ou depois do corpo da saída. Ele tem três subopções: `in_header`, `before_body` e `after_body`. Você precisa conhecer a estrutura básica de um documento HTML ou LaTeX para entender essas opções. A fonte de um documento HTML se parece com isso:

```
<html>

<cabeça>
<!-- conteúdo principal aqui, por exemplo, CSS e JS --&gt;
&lt;/head&gt;

&lt;corpo&gt;
<!-- conteúdo do corpo aqui --&gt; &lt;
body&gt;

&lt;/html&gt;</pre>
```

A opção `in_header` pega um caminho de arquivo e o insere na tag `<head>`. O arquivo `before_body` será inserido logo abaixo da tag de abertura `<body>` e `after_body` será inserido antes da tag de fechamento `</body>`.

Um documento fonte do LaTeX tem uma estrutura semelhante:

```
\documentclass{livro}

% preâmbulo LaTeX
% insira in_header aqui

\begin{documento}
% insira antes_corpo aqui

% de conteúdo corporal aqui

% insira after_body aqui
\end{documento}
```

4.1 Opções YAML

A opção inclui é muito útil e flexível. Para saída HTML, significa que você pode inserir código HTML arbitrário na saída. Por exemplo, quando você tem expressões matemáticas LaTeX renderizadas por meio da biblioteca MathJax na saída HTML e deseja que os números das equações sejam exibidos à esquerda (o padrão é à direita), você pode criar um arquivo de texto que contenha o seguinte código:

```
<script type="text/x-mathjax-config">
MathJax.Hub.Config({
  TeX: { TagSide: "esquerda" } });
</script>
```

Vamos supor que o arquivo tenha o nome `mathjax-number.html` e esteja no diretório raiz do seu livro (o diretório que contém todos os seus arquivos Rmd).

Você pode inserir este arquivo no cabeçalho HTML através da opção `in_header`, por exemplo,

```
...
resultado:
bookdown::gitbook:
incluir:
  in_header: mathjax-number.html
...
```

Outro exemplo é habilitar comentários ou discussões em suas páginas HTML. Existem várias possibilidades, como o Disqus (<https://disqus.com>) ou Hipótese (<https://hypothes.is>). Esses serviços podem ser facilmente incorporados em seu livro HTML por meio da opção `includes` (consulte a Seção 5.5 para obter detalhes).

Da mesma forma, se você estiver familiarizado com o LaTeX, você pode adicionar código LaTeX arbitrário ao preâmbulo. Isso significa que você pode usar qualquer pacote LaTeX e configurar qualquer opção de pacote para o seu livro. Por exemplo, este livro usou a opção `in_header` para usar mais alguns pacotes LaTeX como **booktabs** (para tabelas com melhor aparência) e **longtable** (para tabelas que abrangem

várias páginas) e aplicou uma correção a um problema do XeLaTeX em que os links nos gráficos não funcionam:

```
\usepackage{booktabs}
\usepackage{longtable}

\ifxetex
  \usepackage{letltxmacro}
  \setlength{\XeTeXLinkMargin}{1pt}
  \LetLtxMacro{\SavedIncludeGraphics}{\includegraphics}
  \def\includegraphics#1{%
    \ifx#1\relax\else\#1\fi% #1 captures optional arguments (star/optional arg.)
    \IncludeGraphicsAux{#1}%
  }%
  \newcommand*{\IncludeGraphicsAux}[2]{%
    \XeTeXLinkBox{%
      \SavedIncludeGraphics{#2}%
    }%
  }%
\else

```

O código LaTeX acima é salvo em um arquivo preamble.tex, e os metadados YAML são assim:

```
---
saída:
  bookdown::pdf_book:
    inclui:
      in_header: preâmbulo.tex
---
```

4.2 Temas Às

vezes você pode querer mudar o tema geral da saída, e normalmente isso pode ser feito através da opção in_header de-

4.2 Tema

83

descrito na seção anterior, ou a opção `css` se a saída for HTML. Alguns formatos de saída têm seus temas exclusivos, como `git book`, `tufte_html_book` e `tufte_book2`, e talvez você não queira personalizar muito esses temas. Em comparação, os formatos de saída `html_book()` e `pdf_book()` não estão vinculados a temas específicos e são mais personalizáveis.

Conforme mencionado na Seção 3.1.3, o estilo padrão para `html_book()` é o estilo Bootstrap. O estilo Bootstrap na verdade tem vários temas embutidos que você pode usar, incluindo `default`, `bootstrap`, `cerulean`, `cosmo`, `darkly`, `flatly`, `journal`, `lumen`, `paper`, `readable`, `sandstone`, `simplex`, `spacelab`, `united` e `yeti`. Você pode definir o tema através da opção `tema` , por exemplo,

```
-->
resultado:
bookdown::html_book:
  tema: unidos
-->
```

Se você não gostar de nenhum desses estilos do Bootstrap, você pode definir o tema como nulo e aplicar seu próprio CSS por meio da opção `css` ou `includes` .

Para `pdf_book()`, além da opção `in_header` mencionada na seção anterior, outra possibilidade é alterar a classe do documento. Existem muitas classes LaTeX possíveis para livros, como **memoir** (<https://www.ctan.org/pkg/memoir>), **amsbook** KOMA-Script (<https://www.ctan.org/pkg/koma-script>) e assim por diante. Aqui está uma breve amostra dos metadados YAML que especificam a classe `scrbook` do pacote KOMA-Script:

```
-->
classe de documentos: scrbook
resultado:
bookdown::pdf_book:
  template: null
-->
```

Alguns editores (por exemplo, Springer e Chapman & Hall/CRC) têm seus próprios arquivos de classe ou estilo LaTeX. Você pode tentar alterar a opção documentclass para usar suas classes de documentos, embora normalmente não seja tão simples assim. Você pode acabar usando in_header, ou até mesmo criar um modelo Pandoc LaTeX personalizado para acomodar essas classes de documentos.

Note que quando você mudar documentclass, você provavelmente irá especificar um argumento Pandoc adicional --top-level-division=chapter para que Pandoc saiba que os cabeçalhos de primeiro nível devem ser tratados como capítulos ao invés de seções (este é o padrão quando documentclass é livro), por exemplo,

```
classe de documentos: krantz
resultado:
bookdown::pdf_book:
pandoc_args: --top-level-division=capítulo
```

4.3 Modelos

Quando o Pandoc converte Markdown para outro formato de saída, ele usa um modelo sob o capô. O modelo é um arquivo de texto simples que contém algumas variáveis no formato \$variável\$. Essas variáveis serão substituídas por seus valores gerados pelo Pandoc. Abaixo está um modelo muito breve para saída HTML:

```
<html>
<cabeça>
<title>$title$</title>
</head>

<body>
$body$
</body>
</html>
```

Tem duas variáveis título e corpo. O valor do título vem do

campo title dos metadados YAML e body é o código HTML gerado a partir do corpo do documento de entrada Markdown. Por exemplo, suponha que temos um documento Markdown:

```
---
```

Título: Um bom livro

```
--
```

```
# Introdução
```

```
Este é um livro **bom**!
```

Se usarmos o template acima para gerar um documento HTML, seu código fonte ficará assim:

```
<html>
  <cabeça>
    <title>Um bom livro</title>
  </head>

  <corpo>

    <h1>Introdução</h1>

    <p>Este é um livro <strong>bom</strong> !</p>

  </body>
</html>
```

Os modelos reais de HTML, LaTeX e EPUB são mais complicados, mas a ideia é a mesma. Você precisa saber quais variáveis estão disponíveis:

algumas variáveis são variáveis Pandoc internas e algumas podem ser definidas pelos usuários nos metadados YAML ou passadas da opção de linha de comando -V ou --variable. Algumas variáveis só fazem sentido em formatos de saída específicos, por exemplo, a variável documentclass é usada apenas na saída LaTeX. Consulte a documentação do Pandoc para saber mais sobre

essas variáveis, e você pode encontrar todos os modelos padrão do Pandoc no Reppositório do GitHub <https://github.com/jgm/pandoc-templates>.

Observe que para a saída HTML, o **bookdown** requer alguns tokens de comentário adicionais no modelo, e os explicamos na Seção 3.1.3.

4.4 Configuração

Mencionamos rmd_files na Seção 1.3, e há mais configurações (opcionais) que você pode configurar para um livro em _bookdown.yml¹:

- book_filename: o nome do arquivo Rmd principal, ou seja, o arquivo Rmd que é mesclado de todos os capítulos; por padrão, ele é denominado _main.Rmd.
- delete_merged_file: se deseja excluir o arquivo Rmd principal após o livro é renderizado com sucesso.
- before_chapter_script: um ou vários scripts R a serem executados antes de cada capítulo, por exemplo, você pode querer limpar a área de trabalho antes compilando cada capítulo, neste caso você pode usar rm(list = ls(all = TRUE)) no script R.
- after_chapter_script: semelhante a before_chapter_script e o R script é executado após cada capítulo.
- editar: um link no qual os colaboradores podem clicar para editar o documento fonte Rmd da página atual; isso foi projetado principalmente para o GitHub repositórios, pois é fácil editar arquivos de texto simples arbitrários em GitHub mesmo em repositórios de outras pessoas (se você não tiver acesso ao repositório, o GitHub irá bifurcá-lo automaticamente e deixar você enviar um pull request depois de terminar de editar o arquivo). Esse link deve ter %s nele, que será substituído pelo nome real do arquivo Rmd para cada página.
- histórico: semelhante ao editar, um link para o histórico de edição/commit da página de aluguel.
- view: semelhante a editar, um link para o código fonte da página atual.

¹Para o formato **bs4_book()**, os campos de edição, histórico e visualização não têm efeito e configuração semelhante pode ser especificada com o argumento **repo** da função de saída.

4.5 Internacionalização

87

- `rmd_subdir`: se deve procurar por arquivos Rmd de origem do livro em subdiretórios (por padrão, apenas o diretório raiz é pesquisado). Isso pode ser um booleano (por exemplo, `true` irá procurar por arquivos Rmd de origem do livro no diretório do projeto e todos os subdiretórios) ou lista de caminhos se você quiser procure por arquivos Rmd de origem de livro em um subconjunto de subdiretórios.
- `output_dir`: o diretório de saída do livro (`_book` por padrão); isto configuração é lida e usada por `render_book()`.
- `clean`: um vetor de arquivos e diretórios a serem limpos pelo função `clean_book()`.

Aqui está um exemplo de `_bookdown.yml`:

```
book_filename: "meu-livro.Rmd"
delete_merged_file: true
before_chapter_script: ["script1.R", "script2.R"]
after_chapter_script: "script3.R"
ver: https://github.com/rstudio/bookdown-demo/blob/master/%s
edit: https://github.com/rstudio/bookdown-demo/edit/master/%s
output_dir: "saída do livro"
clean: ["meu-livro.bbl", "R-pacotes.bib"]
```

4.5 Internacionalização

Se o idioma do seu livro não for o inglês, você precisará traduzir algumas palavras e frases em inglês para o seu idioma, como o palavras “Figura” e “Tabela” quando figuras/tabelas são numeradas automaticamente na saída HTML. A internacionalização pode não ser um problema para saída LaTeX, já que alguns pacotes LaTeX podem traduzir automaticamente esses termos para o idioma local, como o pacote `ctexcap` para chinês.

Para saída não LaTeX, você pode definir o campo de idioma no arquivo de configuração `_bookdown.yml`. Atualmente as configurações padrão são:

```
Língua:  

etiqueta:  

figo: 'Figura'  

aba: 'Tabela'  

eq: 'Equação'  

thm: 'Teorema'  

lema : 'lema'  

cor: 'Corolário' prp:  

'Proposição'  

cnj: 'Conjecture'  

def: 'Definição'  

exm: 'Exemplo'  

exr: 'Exercício'  

hyp: 'Hipótese'  

prova: 'Prova.'  

observação: 'Observação.'  

solução: 'Solução.'  

ui:  

editar: editar  

nome_capítulo: "  

apêndice_name: "
```

Por exemplo, se você quiser a FIGURA xx em vez da Figura xx, você pode alterar fig para "FIGURE":

```
Língua:  

etiqueta:  

figo: "FIGURA"
```

Os campos sob ui são usados para especificar alguns termos na interface do usuário. O campo de edição especifica o texto associado ao link de edição em _bookdown.yml (Seção 4.4). Os campos nome_do_capítulo, nome_do_apêndice , fig, tab e eq podem ser uma cadeia de caracteres a ser anexada ao capítulo (por exemplo, 'CAPÍTULO ') ou número de referência (por exemplo, 'FIGURA '), ou uma função R que recebe um número (capítulo

4.5 Internacionalização

89

put e retorna uma string. (por exemplo, `expr function(i) paste('Capítulo', i))`.

Aqui está um exemplo para o húngaro:

Língua:

etiqueta:

`fig: !expr function(i) paste(i, 'figura')`

ui:

`nome_do_capítulo: !expr function(i) paste0(i, 'capítulo.')`

Apenas para nome_do_capítulo e nome_do_apêndice , se for um vetor de caractere de comprimento 2, o prefixo do título do capítulo será `paste0(chapter_name[1], i, nome_do_capítulo[2])`, onde i é o número do capítulo.

Há uma ressalva quando você escreve em uma linguagem que usa multibyte caracteres, como chinês, japonês e coreano (CJK): o Pandoc não pode gerar identificadores de cabeçalhos de seção que sejam caracteres CJK puros, portanto, você não poderá fazer referência cruzada de seções (eles não têm rótulos), a menos que você atribua identificadores manualmente a eles anexando `{#identifier}` ao cabeçalho da seção, onde identificador é um identificador de sua escolha.

5

Editando

Neste capítulo, explicamos como editar, construir, visualizar e veicular o livro localmente. Você pode usar qualquer editor de texto para editar o livro, e mostraremos algumas dicas para usar o RStudio IDE. Apresentaremos as funções subjacentes do R para construir, visualizar e servir o livro antes de apresentarmos o editor, para que você realmente entenda o que acontece nos bastidores quando você clica em um determinado botão no IDE do RStudio e também pode personalizar outros editores chamando essas funções.

5.1 Construir o livro

Para construir todos os arquivos Rmd em um livro, você pode chamar a função `render_book()` em `bookdown`. Abaixo estão os argumentos de `render_book()`:

```
render_book(input = ".", output_format = NULL, ...,
           limpo = VERDADEIRO,
           envir = parent.frame(), clean_envir = interactive(), output_dir = NULL,
           new_session = NA, preview = FALSE, config_file = "_bookdown.yml")
```

O argumento mais importante é `output_format`, que pode receber uma cadeia de caracteres do formato de saída (por exemplo, 'bookdown::gitbook'). Você pode deixar este argumento vazio e o formato de saída padrão será o primeiro formato de saída especificado nos metadados YAML do primeiro arquivo Rmd ou um arquivo YAML separado `_output.yml`, conforme mencionado na Seção 4.4.

Se você planeja gerar vários formatos de saída para um livro, é recomendável especificar todos os formatos em `_output.yml`.

Uma vez que todos os formatos são especificados em `_output.yml`, é fácil escrever um R ou

Shell script ou Makefile para compilar o livro. Abaixo está um exemplo simples de usar um script Shell para compilar um livro em HTML (com o GitBook estilo) e PDF:

```
#!/usr/bin/env Rscript
```

```
bookdown::render_book("index.Rmd", "bookdown::gitbook")
bookdown::render_book("index.Rmd", "bookdown::pdf_book")
```

O script Shell não funciona no Windows (não é verdade, no entanto), mas espero que tenha entendido.

O argumento ... é passado para a função de formato de saída. Argumentos clean e envir são passados para rmarkdown::render(), para decidir se limpe os arquivos intermediários e especifique o ambiente para avaliar o código R, respectivamente.

O diretório de saída do livro pode ser especificado através do argumento output_dir . Por padrão, o livro é gerado no diretório _book . este também pode ser alterado através do campo output_dir no arquivo de configuração _bookdown.yml , para que você não precise especificá-lo várias vezes para renderização de um livro para vários formatos de saída. O argumento new_session foi explicado na Seção 1.4. Quando você define visualização = TRUE , apenas os arquivos Rmd especificados no argumento de entrada são renderizados, o que pode ser conveniente ao visualizar um determinado capítulo, já que você não recompilar o livro inteiro, mas ao publicar um livro, esse argumento certamente deve ser definido como FALSE .

Vários arquivos de saída serão gerados por render_book(). Algumas vezes você pode querer limpar o diretório do livro e começar tudo de novo novamente, por exemplo, remova a figura e os arquivos de cache que foram gerados automaticamente do knitr. A função clean_book() foi projetada para este propósito. Por padrão, ele informa quais arquivos de saída você pode excluir. Se você consultou esta lista de arquivos e tem certeza de que nenhum arquivo foram erroneamente identificados como arquivos de saída (você certamente não quer para excluir um arquivo de entrada que você criou manualmente), você pode excluir todos usando bookdown::clean_book(TRUE). Como a exclusão de arquivos é uma operação relativamente perigosa, recomendamos que você mantenha

seu livro por meio de ferramentas de controle de versão, como GIT, ou um serviço que suporte backup e restauração, para que você não perca determinados arquivos para sempre se excluí-los por engano.

5.2 Visualizar um capítulo

Construir o livro inteiro pode ser lento quando o tamanho do livro é grande.

Duas coisas podem afetar a velocidade de construção de um livro: a computação em pedaços de código R e a conversão de Markdown para outros formatos via Pandoc. O primeiro pode ser melhorado habilitando o cache no **knitr** usando a opção `chunk cache = TRUE`, e não há muito que você possa fazer para tornar o último mais rápido. No entanto, você pode optar por renderizar apenas um capítulo de cada vez usando a função `preview_chapter()` em **bookdown**, e geralmente isso será muito mais rápido do que renderizar o livro inteiro.

Apenas os arquivos Rmd passados para `preview_chapter()` serão renderizados.

A visualização do capítulo atual é útil quando você está focando apenas nesse capítulo, pois você pode ver rapidamente a saída real à medida que adiciona mais conteúdo ou revisa o capítulo. Embora a visualização funcione para todos os formatos de saída, recomendamos que você visualize a saída HTML.

Uma desvantagem de visualizar um capítulo é que as referências cruzadas para outros capítulos não funcionarão, pois o **bookdown** não sabe nada sobre outros capítulos neste caso. Esse é um preço razoavelmente pequeno a pagar pelo ganho de velocidade. Como a visualização de um capítulo apenas renderiza a saída desse capítulo específico, você não deve esperar que o conteúdo de outros capítulos também seja renderizado corretamente. Por exemplo, quando você navega para um capítulo diferente, na verdade você está visualizando a saída antiga desse capítulo (que pode nem existir).

5.3 Sirva o livro

Em vez de executar `render_book()` ou `preview_chapter()` repetidamente, você pode realmente visualizar o livro ao vivo no navegador da web,

e a única coisa que você precisa fazer é salvar o arquivo Rmd. A função `serve_book()` em **bookdown** pode iniciar um servidor web local para servir o Saída HTML baseada no pacote `servr` (Xie, 2021).

```
serve_book(dir = ".", output_dir = "_book", preview = TRUE,
           in_session = TRUE, quiet = FALSE, ...)
```

Você passa o diretório raiz do livro para o argumento `dir`, e isso A função iniciará um servidor web local para que você possa visualizar a saída do livro usando o servidor. A URL padrão para acessar a saída do livro é `http://127.0.0.1:4321`. Se você executar essa função em uma sessão R interativa, essa URL será aberta automaticamente em seu navegador da web. Se você está no RStudio IDE, o RStudio Viewer será usado como o navegador web padrão, então você poderá escrever os arquivos de origem Rmd e visualize a saída no mesmo ambiente (por exemplo, fonte à esquerda e saída à direita).

O servidor escutará as mudanças no diretório raiz do livro: sempre que você modificar qualquer arquivo no diretório do livro, `serve_book()` pode detectar o alterações, recompile os arquivos Rmd e atualize o navegador da Web automaticamente. Se os arquivos modificados não incluem arquivos Rmd, ele apenas atualiza o navegador (por exemplo, se você atualizou apenas um determinado arquivo CSS). este significa que uma vez que o servidor é iniciado, tudo o que você precisa fazer é simplesmente escreva o livro e salve os arquivos. A compilação e a visualização levarão colocar automaticamente à medida que você salva os arquivos.

Se realmente não levar muito tempo para recompilar o livro inteiro, você pode definir o argumento `preview = FALSE`, para que toda vez que você atualizar o livro, o livro inteiro seja recompilado, caso contrário, apenas os capítulos modificados serão recompilados via `preview_chapter()`.

Os argumentos em ... são passados para `servr::httw()`, e por favor consulte sua página de ajuda para ver todas as opções possíveis, como `daemon` e `port`. Lá são prós e contras de usar `in_session = TRUE` ou `FALSE`:

- Para `in_session = TRUE`, você terá acesso a todos os objetos criados no book na sessão R atual: se você usar um servidor daemonizado (através do argumento `daemon = TRUE`), você pode verificar os objetos a qualquer momento quando a sessão R atual não está ocupada; caso contrário, você terá que parar o

servidor antes de poder verificar os objetos. Isso pode ser útil quando você precisa explorar interativamente os objetos R no livro. A desvantagem de `in_session = TRUE` é que a saída pode ser diferente com o livro compilado de uma nova sessão R, porque o estado da sessão R atual pode não ser limpo.

- Para `in_session = FALSE`, você não tem acesso aos objetos no livro da sessão R atual, mas é mais provável que a saída seja reproduzível, pois tudo é criado a partir de novas sessões R. Como essa função é apenas para fins de visualização, a limpeza da sessão R pode não ser uma grande preocupação.

Você pode escolher `in_session = TRUE` ou `FALSE` dependendo de seus casos de uso específicos. Eventualmente, você deve executar `render_book()` a partir de uma nova sessão do R para gerar uma cópia confiável da saída do livro.

5.4 IDE de estúdio

Recomendamos que você atualize¹ seu RStudio IDE se sua versão for inferior a 1.0.0. Conforme mencionado na Seção 1.3, todos os arquivos R Markdown devem ser codificados em UTF-8. Isso é importante especialmente quando seus arquivos contêm caracteres multibyte. Para salvar um arquivo com a codificação UTF-8, você pode usar o menu Arquivo -> Salvar com codificação e escolher UTF-8.

Quando você clica no botão Knit para compilar um documento R Markdown no RStudio IDE, a função padrão chamada pelo RStudio é `rmark down::render()`, que não é o que queremos para livros. Para chamar a função `bookdown::render_book()` em vez disso, você pode definir o campo do site como `bookdown::bookdown_site` nos metadados YAML do documento R Markdown `index.Rmd`, por exemplo,

```
--  
título: "Um bom livro"  
site: bookdown::bookdown_site  
resultado:
```

¹ <https://www.rstudio.com/products/rstudio/download/>

bookdown::gitbook: default

Quando você definir site: bookdown::bookdown_site em index.Rmd, o RStudio poderá descobrir o diretório como um diretório de origem do livro,² e você verá um botão Build Book no painel Build . Você pode clicar no botão para construir o livro inteiro em diferentes formatos, e se você clicar no Knit na barra de ferramentas, o RStudio irá visualizar automaticamente o capítulo atual, e você não precisa usar preview_chapter() explicitamente.

O pacote **bookdown** vem com alguns addins para o RStudio. Se você não estiver familiarizado com os suplementos do RStudio, você pode conferir a documentação em <http://rstudio.github.io/rstudioaddins/>. Depois de instalar o pacote **bookdown** e usar o RStudio v0.99.878 ou posterior, você verá um menu suspenso na barra de ferramentas chamado "Addins" e menu itens como "Preview Book" e "Input LaTeX Math" depois de abrir o cardápio.

O addin "Preview Book" chama bookdown::serve_book() para compilar e servir o livro. Ele bloqueará sua sessão atual do R, ou seja, quando serve_book() está em execução, você não poderá fazer nada no R console mais. Para evitar bloquear a sessão do R, você pode daemonizar o servidor usando bookdown::serve_book(daemon = TRUE). Observe que este addin deve ser usado quando o documento atual aberto no RStudio é no diretório raiz do seu livro, caso contrário, serve_book() pode não ser capaz de encontrar a fonte do livro.

O addin "Input LaTeX Math" é essencialmente um pequeno aplicativo Shiny que fornece uma caixa de texto para ajudá-lo a digitar expressões matemáticas em LaTeX (Figura 5.1). Conforme você digita, você verá a visualização da expressão matemática e seu código-fonte LaTeX. Isso tornará muito menos propenso a erros para digitar expressões matemáticas — quando você digita uma expressão matemática longa em LaTeX sem visualização, é fácil cometer erros como X_ij quando você quis dizer X_{ij}, ou omitindo um colchete de fechamento. Se você selecionou uma expressão matemática LaTeX no editor RStudio antes de clicar o addin, a expressão será carregada e renderizada automaticamente

²Este diretório deve ser um projeto RStudio.

na caixa de texto. Este suplemento foi construído em cima da biblioteca MathQuill (<http://mathquill.com>). Não se destina a fornecer suporte completo a todos os comandos do LaTeX para expressões matemáticas, mas deve ajudá-lo a digitar alguns expressões matemáticas comuns.

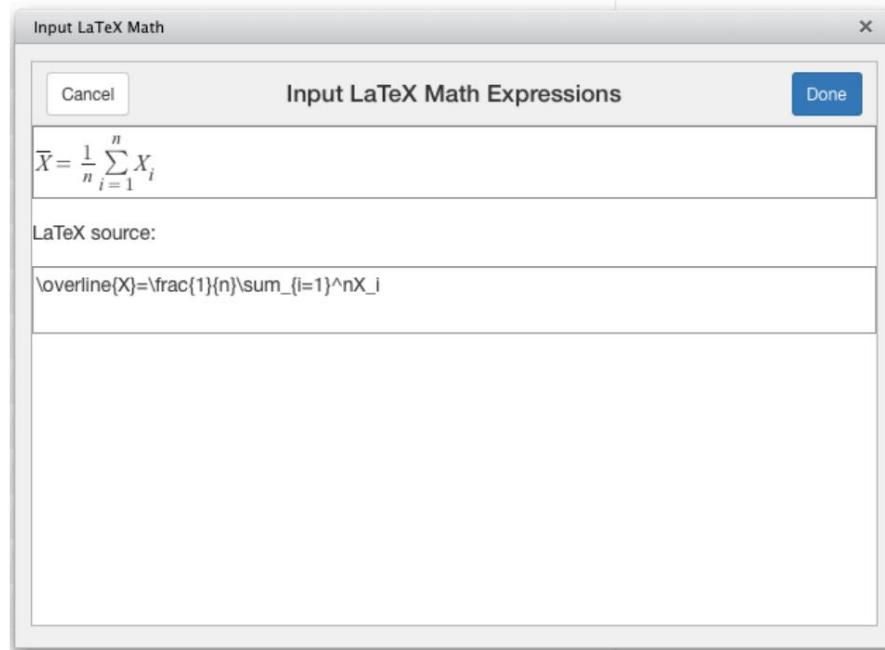


FIGURA 5.1: O suplemento RStudio para ajudar a inserir matemática LaTeX.

Há também outros pacotes R que fornecem suplementos para ajudá-lo a criar livros. O pacote `citr` ([Aust, 2019](#)) fornece um addin chamado “Inserir citações”, que facilita a inserção de citações em documentos R Mark down. Ele verifica seus bancos de dados bibliográficos e mostra todos os itens de citação em um menu suspenso, para que você possa escolher na lista sem lembrar qual chave de citação corresponde a qual item de citação (Figura 5.2).

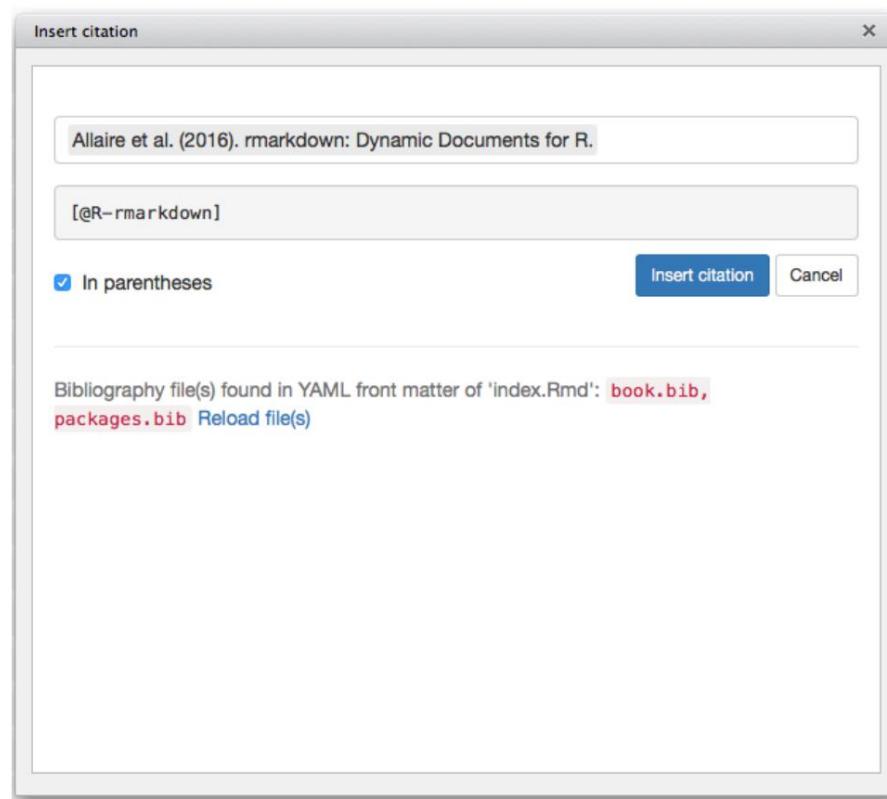


FIGURA 5.2: O complemento do RStudio para ajudar a inserir citações.

5.5 Colaboração

Escrever um livro quase certamente envolverá mais do que uma única pessoa.

Você pode ter coautores e leitores que lhe dão feedback de tempo ao tempo.

Como todos os capítulos de livros são arquivos de texto simples, eles são perfeitos para ferramentas de controle de versão, o que significa que se todos os seus coautores e colaboradores tiverem conhecimento básico de uma ferramenta de controle de versão como o GIT, você poderá colaborar com eles no conteúdo do livro usando essas ferramentas. Na verdade, colaboração com o GIT é possível mesmo que eles não saibam como usar GIT, porque o GitHub tornou possível criar e editar arquivos em

linha diretamente no seu navegador da web. Apenas uma pessoa precisa estar familiarizada com o GIT, e essa pessoa pode configurar o repositório de livros. Os demais colaboradores podem contribuir com conteúdo online, embora tenham mais liberdade se souberem o uso básico do GIT para trabalhar localmente.

Os leitores podem contribuir de duas maneiras. Uma maneira é contribuir com conteúdo diretamente, e a maneira mais fácil é por meio de pull requests³ do GitHub se a fonte do seu livro estiver hospedada no GitHub. Basicamente, qualquer usuário do GitHub pode clicar no botão editar na página de um arquivo de origem Rmd, editar o conteúdo e enviar as alterações para sua aprovação. Se você estiver satisfeito com as alterações propostas (você pode ver claramente o que exatamente foi alterado), você pode clicar no botão "Mesclar" para mesclar as alterações. Se você não estiver satisfeito, pode fornecer seu feedback na solicitação pull, para que o leitor possa revisá-lo ainda mais de acordo com suas necessidades. Mencionamos o botão de edição no estilo GitBook na Seção 3.1.1. Esse botão está vinculado à fonte Rmd de cada página e pode orientá-lo a criar a solicitação pull. Não há necessidade de escrever e-mails para comunicar alterações simples, como corrigir um erro de digitação.

Outra forma de os leitores contribuirem para o seu livro é deixar comentários. Os comentários podem ser deixados de várias formas: e-mails, GitHubissues ou comentários de página HTML. Aqui usamos Disqus (veja Seção 4.1) como exemplo. Disqus é um serviço para incorporar uma área de discussão em suas páginas da web e pode ser carregado via JavaScript. Você pode encontrar o código JavaScript depois de se registrar e criar um novo fórum no Disqus, que se parece com isso:

```
<div id="disqus_thread"></div> <script>

(function() { // NÃO EDITE ABAIXO DESTA LINHA var d =
document, s = d.createElement('script');
s.src = '//yihui.disqus.com/embed.js'; s.setAttribute('data-
timestamp', +new Date());
(d.head || d.body.appendChild(s)); })(); </script>
```

³ <https://help.github.com/articles/about-pull-requests/>

100

5 Edição

```
<noscript> Ative o JavaScript para visualizar o
<a href="https://disqus.com/?ref_noscript">
comentários alimentados por Disqus.</a></noscript>
```

Observe que você precisará substituir o nome yihui pelo seu próprio nome de fórum (este nome deve ser fornecido quando você cria um novo fórum Disqus). Você pode salvar o código em um arquivo HTML chamado, por exemplo, disqus.html. Em seguida, você pode incorporá-lo no final de cada página por meio da opção after_body (a Figura 5.3 mostra como é a área de discussão):

resultado:

```
bookdown::gitbook:
incluir:
after_body: disqus.html
```

In short, you just prepare a few R Markdown book chapters, and **bookdown** can help you turn them into a beautiful book.

The screenshot shows a Disqus comment section integrated into a bookdown page. At the top, it says "0 Comments" and "Yihui Xie". Below that are "Recommend" and "Share" buttons. On the right, there's a "Sort by Best" dropdown. A user comment from "yihui" is displayed, reading "Hi, I want to provide some feedback to this chapter." There's a "Post as Yihui Xie" button at the bottom right of the comment area.

FIGURA 5.3: Uma página de livro com uma área de discussão.

6

Publicação

À medida que desenvolve o livro, você disponibiliza o rascunho do livro ao público para obter feedback antecipado dos leitores, por exemplo, publicá-lo em um site. Depois de terminar de escrever o livro, você precisa pensar em opções para publicá-lo formalmente como cópias impressas ou e-books.

6.1 RStudio Connect

Em teoria, você pode renderizar o livro sozinho e publicar o resultado onde quiser. Por exemplo, você pode hospedar os arquivos HTML em seu próprio servidor web. Fornecemos uma função `publish_book()` em **bookdown** para simplificar o upload do seu livro para <https://bookdown.org>, que é um site fornecido pelo RStudio para hospedar seu livros de graça. Este site é construído em cima do “RStudio Connect”,¹ um Produto RStudio que permite implantar uma variedade de aplicativos relacionados ao R em um servidor, incluindo documentos R Markdown, aplicativos Shiny, gráficos R e assim por diante.

Você não precisa saber muito sobre o RStudio Connect para publicar seu livro para bookdown.org. Basicamente você se inscreve em <https://bookdown.org/conectar/>, e na primeira vez que você tentar executar `bookdown::publish_book()`, ser-lhe-á pedido que autorize o **bookdown** a publicar na sua conta book down.org. No futuro, você simplesmente chama `publish_book()` novamente e o **bookdown** não pedirá mais nada.

¹<https://www.rstudio.com/products/connect/>

```
publish_book(nome = NULL, conta = NULL, servidor = NULL,
            render = c("nenhum", "local", "servidor"))
```

O único argumento de `publish_book()` que você pode querer tocar é `render`. Ele determina se você deseja renderizar o livro antes de publicá-lo. Se você executou `render_book()` antes, não precisa alterar este argumento, caso contrário, você pode configurá-lo como 'local':

```
bookdown::publish_book(render = "local")
```

Se você configurou seu próprio servidor RStudio Connect, certamente pode publicar o livro nesse servidor em vez de `bookdown.org`.

6.2 Netlify Solte

Netlify (<https://netlify.com>) é uma plataforma que oferece hospedagem em nuvem e serviços de back-end sem servidor para sites estáticos. A Netlify oferece níveis de serviço gratuitos e pagos, mas também oferece um serviço chamado Netlify Drop (<https://app.netlify.com/drop>), que é uma opção de publicação gratuita que não requer uma conta Netlify para iniciar. Esta opção não depende de seu projeto `bookdown` estar em um repositório com controle de versão. Tudo que você precisa é de um projeto `bookdown` que você pode construir localmente.

6.2.1 A sequência de pipeline de construção e

implantação Esta abordagem de publicação configura o seguinte fluxo de eventos:

1. Você começa com um projeto de `bookdown` local .
2. Você cria seu livro localmente em um diretório de saída de sua escolha (`_book/` por padrão).
3. Você acessa o Netlify Drop (<https://app.netlify.com/drop>), e arraste e solte o diretório de saída na interface de usuário baseada no navegador Netlify.

4. Você faz alterações em seu livro, reconstrói localmente e depois arrasta e solta o diretório de saída novamente no Netlify para atualizar.

O acima é uma visão geral - continue lendo para obter instruções passo a passo.

6.2.2 Antes de começar

Comece com um projeto de **bookdown** local . Ele não precisa estar no GitHub ou em outro repositório com controle de versão.

Se você não tiver um livro existente, você pode criar um livro HTML simples **para** praticar. Veja a Figura 3.3 para saber como criar um novo livro no RStudio, ou use a função `bookdown::create_gitbook()` ou `bookdown::create_bs4_book()` do seu console R se você não usar o RStudio.

6.2.3 Construa seu livro

A partir de seu projeto **bookdown** , construa seu livro localmente usando qualquer método do Capítulo 5.1 que você preferir.

6.2.4 Implante seu site Vá

para Netlify Drop ([netlify.com/drop2](https://app.netlify.com/drop2)), onde você deverá ver uma caixa que informa para “Arraste e solte a pasta do seu site aqui”.

Em seguida, arraste e solte o diretório de saída do seu projeto **bookdown** (_book/ por padrão, a menos que você tenha alterado isso em seu arquivo _bookdown.yml) nessa caixa em seu navegador da web. Você deve ver seu livro ser implantado rapidamente com um nome de subdomínio aleatório no formato <https://random word-12345.netlify.com>.

Você também verá um aviso informando que os sites não reivindicados são excluídos após 24 horas. Você pode se inscrever em uma conta Netlify para reivindicar seu site e mantê-lo online permanentemente.

² <https://app.netlify.com/drop>

6.2.5 Opcional: atualize seu site

Depois de se inscrever no Netlify, você *pode* atualizar esse tipo de site, mas é uma atualização manual. Vá para Netlify.com e navegue para encontrar seu site e clique em “Implantações”. Você deverá ver uma caixa conforme mostrado na Figura 6.1, indicando que você pode arrastar e soltar a pasta do seu site para atualizar seu site (talvez seja necessário rolar até o final desta página).

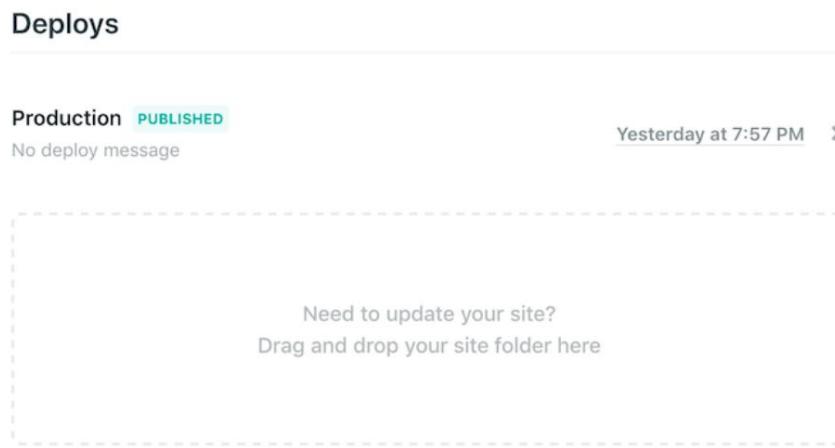


FIGURA 6.1: Captura de tela da caixa de atualização de implantação de arrastar e soltar no Netlify.

Edite seu livro, crie-o localmente novamente e arraste e solte o diretório de saída aqui novamente.

6.2.6 Opcional: altere o subdomínio padrão

Navegue até a página de destino do seu site em Netlify.com (<https://app.netlify.com>), clique em *Visão geral > Configurações do site*. Em *Informações do site*, clique em *Alterar nome do site* e atualize-o para o nome desejado. Se você quiser usar seu próprio domínio em vez do subdomínio da Netlify, leia a documentação em <https://docs.netlify.com/domains/https/custom-domains/>.

6.2.7 Desvantagens e alternativas

Esse fluxo de trabalho é ótimo para compartilhar rapidamente um protótipo de livro. No entanto, se você optar por não reivindicar seu site, o link expirará em 24 horas. Mesmo que você reivindique seu site e configure uma conta Netlify, esse fluxo de trabalho não é ideal para livros nos quais você está editando ou colaborandoativamente, porque toda vez que você atualiza sua versão local do livro, você precisa fazer o upload manual do livro para Netlify . Você também não está colhendo os benefícios do controle de versão com essa abordagem.

6.3 GitHub

Você pode hospedar seu livro no GitHub gratuitamente através do GitHub Pages (<https://pages.github.com>). O GitHub suporta Jekyll (<http://jekyllrb.com>), um construtor de sites estático, para criar um site a partir de arquivos Markdown. Esse pode ser o caso de uso mais comum do GitHub Pages, mas o GitHub também suporta arquivos HTML estáticos arbitrários, então você pode hospedar os arquivos de saída HTML do seu livro no GitHub. A chave é criar um arquivo oculto `.nojekyll` que informe ao GitHub que seu site não deve ser construído via Jekyll, já que a saída HTML do livro já é um site independente.

```
# suponha que você tenha inicializado o repositório git,
# e estão sob o diretório do repositório de livros agora

# cria um arquivo oculto .nojekyll
touch .nojekyll
# adicione ao git aqui porque não aparecerá no RStudio git add .nojekyll
```

Se você estiver no Windows, pode não ter o comando `touch` e pode criar o arquivo em R usando `file.create('.nojekyll')`.

Uma abordagem é publicar seu livro como um site do GitHub Pages a partir de uma pasta `/docs` em seu branch master , conforme descrito na Ajuda do GitHub.³ Primeiro,

³<http://bit.ly/2cvloKV>

defina o diretório de saída do seu livro como /docs adicionando a linha output_dir: "docs" para o arquivo de configuração _bookdown.yml. Então depois empurrando suas alterações para o GitHub, vá para as configurações do seu repositório e em "GitHub Pages" altere a "Source" para "master branch /docs" pasta". Neste caso, o arquivo .nojekyll deve estar na pasta /docs .

Uma abordagem alternativa é criar uma ramificação gh-pages em seu repositório, construir o livro, colocar a saída HTML (incluindo todos os recursos externos como imagens, CSS e arquivos JavaScript) nesta ramificação e enviar a ramificação para o repositório remoto. Se o seu repositório de livros não tiver o branch gh-pages , você pode usar os seguintes comandos para criar comeu um:

```
# suponha que você tenha inicializado o repositório git,
# e estão sob o diretório do repositório de livros agora

# cria um branch chamado gh-pages e limpa tudo
git checkout --orphan gh-pages
git rm -rf .

# cria um arquivo oculto .nojekyll
toque em .nojekyll
git add .nojekyll

git commit -m"Commit inicial"
git push origin gh-pages
```

Depois de configurar o GIT, o restante do trabalho pode ser automatizado por meio de um script (Shell, R ou Makefile, dependendo de sua preferência). Basicamente, você compila o livro para HTML e, em seguida, executa comandos git para enviar o arquivos para o GitHub, mas você provavelmente não quer fazer isso repetidamente novamente manualmente e localmente. Pode ser muito útil automatizar o processo de publicação completamente na nuvem, portanto, uma vez configurado corretamente, tudo o que você precisa fazer é escrever o livro e enviar os arquivos de origem Rmd para o GitHub, e seu livro sempre será criado e publicado automaticamente do lado do servidor.

Um serviço que você pode utilizar é o Travis CI (<https://travis-ci.com>). Isto

é gratuito para repositórios públicos no GitHub e foi projetado para integração contínua (CI) de pacotes de software. O Travis CI pode ser conectado ao GitHub no sentido de que sempre que você enviar para o GitHub, o Travis pode ser acionado para executar determinados comandos/scripts na versão mais recente do seu repositório.⁴ Esses comandos são especificados em um arquivo YAML chamado `.travis.yml` em o diretório raiz do seu repositório, e eles geralmente são para fins de teste de software, mas na verdade eles são bastante abertos, o que significa que você pode executar comandos arbitrários em uma máquina Travis (virtual). Isso significa que você certamente pode executar seus próprios scripts para construir seu livro no Travis. Observe que o Travis suporta apenas Ubuntu e Mac OS X no momento, então você deve ter algum conhecimento básico sobre comandos Linux/Unix.

A próxima pergunta é, como publicar o livro construído no Travis no GitHub? Basicamente, você precisa conceder ao Travis acesso de gravação ao seu repositório GitHub. Essa autorização pode ser feita de várias maneiras, e a mais fácil para iniciantes pode ser um token de acesso pessoal. Aqui estão alguns passos que você pode seguir:

1. Crie um token de acesso pessoal⁵ para sua conta no GitHub (certifique-se de habilitar o escopo “repo” para que o uso desse token permita a gravação em seus repositórios do GitHub).
2. Criptografe-o na variável de ambiente GITHUB_PAT via linha de comando `travis encrypt` e armazene-o em `.travis.yml`, por exemplo, `travis encrypt GITHUB_PAT=TOKEN`. Se você não sabe como instalar ou usar a ferramenta de linha de comando Travis, basta salvar esta variável de ambiente em <https://travis-ci.com/user/repo/settings> em que user é seu ID do GitHub e repo é o nome do repositório.
3. Você pode clonar este branch gh-pages no Travis usando seu token GitHub, adicionar os arquivos de saída HTML do R Markdown (não se esqueça de adicionar figuras e arquivos de estilo CSS também) e enviar para o repositório remoto.

⁴Você precisa autorizar o serviço Travis CI para seu repositório no GitHub primeiro. Veja <https://docs.travis-ci.com/user/getting-started/> para saber como começar com o Travis CI. ⁵ <http://bit.ly/2cEBYWb>

Suponha que você esteja no branch master agora (onde você coloca os arquivos de origem Rmd) e compilou o livro no diretório _book . O que você pode fazer a seguir no Travis é:

```
# configure seu nome e e-mail se ainda não o fez git config --global
user.email "you@example.com"
git config --global user.name "Seu nome"

# clona o repositório para o diretório de saída do livro git clone -b gh-
pages \
https://${GITHUB_PAT}@github.com/${TRAVIS_REPO_SLUG}.git \ book-output

cd livro-saída
git rm -rf *
cp -r ./_book/* .
git add --all *
git commit -m "Atualizar o livro"
git push -q origin gh-pages
```

O nome da variável GITHUB_PAT e o nome do diretório book-output são arbitrários, e você pode usar qualquer nome que preferir, desde que os nomes não entrem em conflito com nomes de variáveis de ambiente ou nomes de diretórios existentes. Este script, junto com o script de compilação que mencionamos na Seção 5.1, podem ser colocados no branch master como scripts Shell, por exemplo, você pode nomeá-los como _build.sh e _deploy.sh. Então seu .travis.yml pode ficar assim:

```
idioma: r
pandoc_version: 1.19.2.1

ambiente:

global:
  - seguro: A_LONG_ENCRYPTED_STRING

script_antes:
  - chmod +x ./_build.sh
```

```
- chmod +x ./_deploy.sh
```

roteiro:

```
- ./_build.sh
- ./_deploy.sh
```

A chave de idioma diz ao Travis para usar uma máquina virtual que tenha o R instalado. A chave segura é seu token de acesso pessoal criptografado. Se você já salvou a variável GITHUB_PAT usando a interface da web no Travis em vez da ferramenta de linha de comando travis encrypt, você pode deixar essa chave de fora.

Como este serviço Travis é principalmente para verificar pacotes R, você também precisará de um arquivo DESCRIPTION (falso) como se o repositório do livro fosse um pacote R. A única coisa neste arquivo que realmente importa é a especificação das dependências. Todas as dependências serão instaladas através do pacote **de vtools**. Se uma dependência for CRAN ou BioConductor, você pode simplesmente listá-la no campo Imports do arquivo DESCRIPTION . Se estiver no GitHub, você pode usar o campo Remotes para listar o nome do repositório. Ser baixo é um exemplo:

Pacote: espaço reservado

Tipo: livro

Título: Não importa.

Versão: 0.0.1

Importações: bookdown, ggplot2

Controles remotos: rstudio/bookdown

Se você usa a infraestrutura baseada em contêiner⁶ no Travis, pode habilitar o cache usando sudo: false em .travis.yml. Normalmente você deve armazenar em cache pelo menos dois tipos de diretórios: o diretório figure (por exemplo, _main_files) e o diretório de cache (por exemplo, _main_cache). Esses nomes de diretório também podem ser diferentes se você tiver especificado as opções do knitr chunk fig.path e cache.path, mas eu recomendo fortemente que você não altere essas opções. Os diretórios de figura e cache são

⁶ <https://docs.travis-ci.com/user/workers/container-based-infrastructure/>

armazenado no diretório `_bookdown_files` do diretório raiz do livro.

Um arquivo `.travis.yml` que habilitou o cache da figura `knitr` e dos diretórios de cache pode ter configurações adicionais `sudo` e `cache` como esta:

```
sudo: falso

cache:
  packages: sim
  diretórios:
    - $TRAVIS_BUILD_DIR/_bookdown_files
```

Se o seu livro é muito demorado para construir, você pode usar as configurações acima no Travis para economizar tempo. Observe que `packages: yes` significa que os pacotes R instalados no Travis também são armazenados em cache.

Todos os scripts e configurações acima podem ser encontrados no repositório `bookdown-demo` : <https://github.com/rstudio/bookdown-demo/>. Se você copiá-los para seu próprio repositório, lembre-se de alterar a chave segura em `.travis.yml` usando sua própria variável criptografada `GITHUB_PAT`.

GitHub e Travis CI certamente não são as únicas opções para construir e publicar seu livro. Você é livre para armazenar e publicar o livro em seu próprio servidor.

6.4 Recursos para publicação em HTML

6.4.1 Páginas HTML 404

Se um leitor tentar acessar uma página em seu livro que não pode ser encontrada, um navegador exibirá um erro 404⁷, pois não pode encontrar a página da Web solicitada. Este erro 404 é exibido em uma página 404. Cada servidor web tem um padrão para uma página 404. No entanto, a maioria das plataformas de serviço da Web, como Netlify, Github Pages e Gitlab Pages, usará um arquivo chamado `404.html` na raiz do seu site como uma página de erro personalizada, se você o fornecer.

Para todos os formatos de livro HTML, `bookdown` cria um `404.html` personalizado em

⁷ https://en.wikipedia.org/wiki/HTTP_404

6.4 Recursos para publicação em HTML

seu diretório de saída usando conteúdo simples (um cabeçalho e um corpo de 2 parágrafos); veja a Figura 6.2.

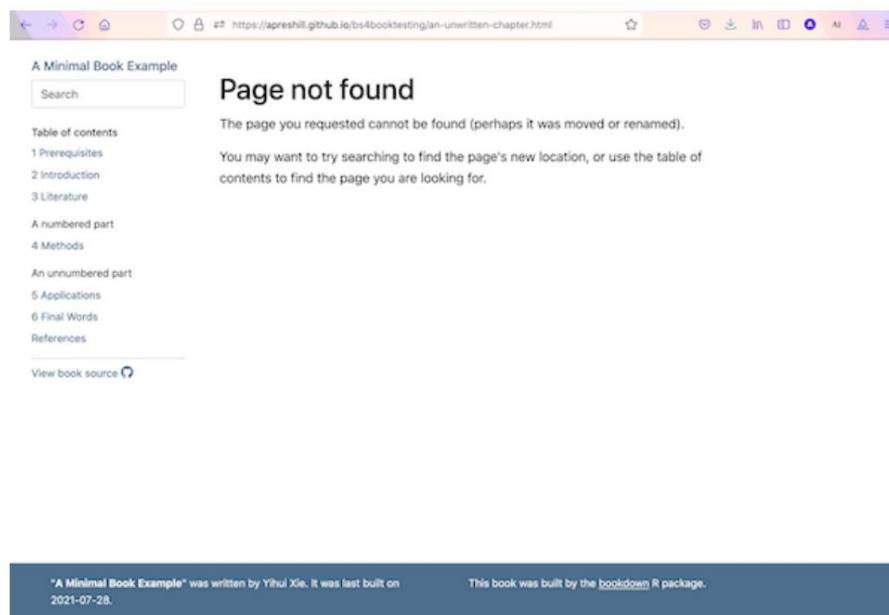


FIGURA 6.2: Captura de tela de uma página 404 de exemplo.

Como você pode ver, esta página 404 está incorporada ao livro para que os leitores podem encontrar rapidamente o caminho de volta ao conteúdo do livro. A estrutura geral do site do livro (incluindo barra de navegação, rodapé, barras laterais, etc.) e o estilo CSS são preservados na página 404.

Para personalizar a página 404 em vez de usar o único **livro fornecido**, você pode adicionar um arquivo `_404.Rmd` ou `_404.md` ao seu projeto raiz. Se um dos arquivos for encontrado quando você renderizar o livro, o conteúdo será ser renderizado e incluído como o corpo da página 404 incorporada a estrutura do livro.

Se um arquivo `404.html` já existir no repositório principal no nível raiz (ao lado dos arquivos `.Rmd` do livro), então o **bookdown** deixará esse arquivo como está e não irá sobrescrevê-lo. Isso ocorre porque assumimos que você já tem um mecanismo em vigor em seu fluxo de trabalho de publicação para usar esse `404.html`.

6.4.2 Metadados para compartilhamento

Bookdown Livros HTML fornecerão metadados HTML para compartilhamento social em plataformas como Twitter, Facebook e LinkedIn, usando as informações fornecidas no index.Rmd YAML. Para configurar, defina o URL do seu livro e o caminho para o arquivo de imagem de capa. O caminho pode ser para um URL absoluto ou para um arquivo de imagem relativo localizado em seu projeto. Sua o título e a descrição do livro também são usados. Um bom efeito de definir esses opções é que quando os leitores compartilharem o link do seu livro em sites de redes sociais, o link será automaticamente expandido para um cartão com a imagem da capa e a descrição do livro.

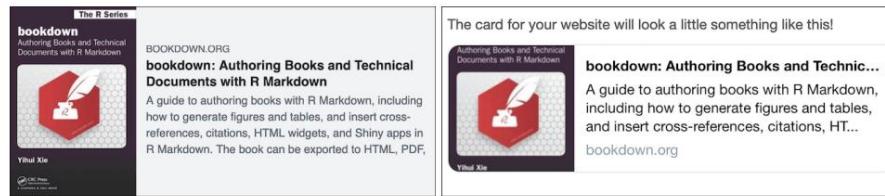


FIGURA 6.3: Capturas de tela mostrando a imagem da capa, título e descrição de um livro HTML quando o link é compartilhado no Facebook e LinkedIn (esquerda) e no Twitter (direita).

Seja qual for o método usado para publicar seu livro HTML, você pode verificar seus metadados usando <https://www.opengraph.xyz>, que mostra pré-visualizações de como seu link ficará quando compartilhado entre plataformas. Você também pode usar uma ferramenta de desenvolvedor específica da plataforma:

- Facebook: <https://developers.facebook.com/tools/debug/>
- LinkedIn: <https://www.linkedin.com/post-inspector/>
- Twitter: <https://cards-dev.twitter.com/validator>

6.5 Editores

Além de publicar seu livro online, você certamente pode considerar publicá-lo com uma editora. Por exemplo, este livro foi publicado com Chapman & Hall/CRC, e há também uma versão online gratuita em <https://bookdown.org/yihui/bookdown/> (com acordo com o

editor). Outra opção que você pode considerar é a autopublicação (<https://en.wikipedia.org/wiki/Self-publishing>) se você não quiser trabalhar com uma editora estabelecida. Pablo Casas escreveu duas postagens de blog que podem ser úteis: “Como autopublicar um livro”⁸ e “Como publicar um livro por conta própria: personalizando o bookdown”⁹.

Será muito mais fácil publicar um livro escrito com **bookdown** se a editora que você escolher suportar o LaTeX. Por exemplo, Chapman & Hall fornece uma classe LaTeX chamada krantz.cls, e Springer fornece sv mono.cls. Para aplicar essas classes LaTeX ao seu livro PDF, defina a classe do documento nos metadados YAML de index.Rmd para o nome da classe (sem a extensão .cls).

A classe LaTeX é a configuração mais importante nos metadados YAML. Ele controla o estilo geral do livro PDF. Muitas vezes, existem outras configurações que você deseja ajustar, e mostraremos alguns detalhes sobre este livro abaixo.

Os metadados YAML deste livro contêm estas configurações:

```
classe de documentos: krantz
muito: sim
lof: sim
tamanho da fonte: 12pt
monofont: "Código Fonte Pro"
monofontoptions: "Escala=0,7"
```

O campo lot: yes significa que queremos a Lista de Tabelas e, da mesma forma, lof significa Lista de Figuras. O tamanho da fonte base é 12pt, e usamos o Source Code Pro¹⁰ como a fonte monoespaçada (largura fixa), que é aplicada a todo o código do programa neste livro.

No preâmbulo do LaTeX (Seção 4.1), temos mais algumas configurações. Primeiro,

⁸ <https://blog.datascienceheroes.com/how-to-self-publish-a-book/>

⁹ <https://blog.datascienceheroes.com/how-to-self-publish-a-book-customizing>

livro/

¹⁰ <https://www.fontsquirrel.com/fonts/source-code-pro>

definimos a fonte principal como Alegreya¹¹ e, como essa fonte não possui o recurso SMALL CAPITALS, usamos a fonte Alegreya SC.

```
\setmainfont[
    UprightFeatures={SmallCapsFont=AlegreyaSC-Regular}
]{Alegreya}
```

Os comandos a seguir tornam os ambientes flutuantes menos propensos a flutuar, permitindo que eles ocupem frações maiores de páginas sem flutuar.

```
\renewcommand{\textfraction}{0,05}
\renewcommand{\topfraction}{0,8}
\renewcommand{\bottomfraction}{0,8}
\renewcommand{\floatpagefraction}{0,75}
```

Como o krantz.cls fornecia um ambiente VF para cotações, redefinimos o ambiente de cotação padrão para VF. Você pode ver seu estilo na Seção 2.1.

```
\renewenvironment{quote}{\begin{VF}}{\end{VF}}
```

Em seguida, redefinimos os hiperlinks como notas de rodapé, pois quando o livro é impresso em papel, os leitores não conseguem clicar nos links no texto. As notas de rodapé dirão a eles quais são os links reais.

```
\let\oldhref\href
\renewcommand{\href}[2]{\#2\footnote{\url{\#1}}}
```

Também temos algumas configurações para o formato bookdown::pdf_book em _out put.yml:

```
bookdown::pdf_book:
  inclui:
```

¹¹<https://www.fontsquirrel.com/fonts/alegreya>

6.5 Editores

115

```

in_header: latex/preamble.tex
before_body: latex/before_body.tex
after_body: latex/after_body.tex
keep_tex: sim
dev: "cairo_pdf"
latex_engine: xelatex
citation_package: natbib
template: null
pandoc_args: --top-level-division=capítulo
toc_unnumbered: não
toc_appendix: sim
quote_footer: ["\\VA{", "}{"] highlight_bw:
sim

```

Todas as configurações de preâmbulo que mencionamos acima estão no arquivo `la`
`tex/preamble.tex`, onde também especificamos que o front
começa:

```
\frontmatter
```

Em `latex/before_body.tex`, inserimos algumas páginas em branco exigidas pela
editora e escrevemos a página de dedicatória. Antes do primeiro capítulo do livro,
inserimos

```
\matéria principal
```

para que o LaTeX saiba como alterar o estilo de numeração de página de algarismos
romanos (para a frente) para algarismos arábicos (para o corpo do livro).

Imprimimos o índice em `latex/after_body.tex` (Seção 2.9).

O dispositivo gráfico (dev) para salvar plotagens foi configurado para `cairo_pdf` para
que as fontes sejam incorporadas nas plotagens, já que o dispositivo padrão `pdf` não
incorpora fontes. É provável que seu editor de texto exija que você incorpore todas
as fontes usadas no PDF, para que o livro possa ser impresso exatamente como
parece, caso contrário, algumas fontes podem ser substituídas e o tipo de letra pode
ser imprevisível.

O campo quote_footer era para garantir que os rodapés das citações estivessem alinhados à direita: o comando LaTeX \VA{} foi fornecido pelo krantz.cls para incluir o rodapé das citações.

A opção highlight_bw foi configurada para true para que os blocos de código destacados de sintaxe de cores em destaque fossem convertidos em escala de cinza, pois este livro será impresso em preto e branco.

O livro foi compilado em PDF através do xelatex para facilitar para nós para usar fontes personalizadas.

Todas as configurações acima, exceto o ambiente VF e o comando \VA{} pode ser aplicado a qualquer outra classe de documento LaTeX.

Caso você queira trabalhar também com a Chapman & Hall, você pode comece com a cópia de krantz.cls em nosso repositório ([https://github.com/rstudio/bookdown/tree/master\(inst/examples\)](https://github.com/rstudio/bookdown/tree/master(inst/examples))) em vez da cópia que você obter de seu editor. Trabalhamos com o suporte técnico do LaTeX para corrigir alguns problemas com esta classe LaTeX, então espero que funcione bem para o seu livro se você usar **bookdown**.

UMA*Ferramentas de software*

Para aqueles que não estão familiarizados com os pacotes de software necessários para usar o R Markdown, damos uma breve introdução à instalação e manutenção desses pacotes.

A.1 Pacotes R e R

O R pode ser baixado e instalado a partir de qualquer espelho CRAN (Comprehensive R Archive Network), por exemplo, <https://cran.rstudio.com>.

Observe que haverá alguns novos lançamentos do R todos os anos, e você pode querer atualizar o R ocasionalmente.

Para instalar o pacote **bookdown**, você pode digitar isso em R:

```
install.packages("bookdown")
```

Isso instala todos os pacotes R necessários. Você também pode optar por instalar todos os pacotes opcionais, se não se importar muito se esses pacotes serão realmente usados para compilar seu livro (como **htmlwidgets**):

```
install.packages("bookdown", dependencies = TRUE)
```

Se você quiser testar a versão de desenvolvimento do **bookdown** no GitHub, primeiro você precisa instalar o **devtools**:

```
if (!requireNamespace("devtools")) install.packages("devtools") devtools::install_github("rstudio/bookdown")
```

Os pacotes R também são constantemente atualizados no CRAN ou no GitHub, então você pode querer atualizá-los de vez em quando:

```
update.packages(ask = FALSE)
```

Embora não seja obrigatório, o RStudio IDE pode tornar muitas coisas muito mais fáceis quando você trabalha em projetos relacionados ao R. O RStudio IDE pode ser baixado em <https://www.rstudio.com>.

A.2 Pandoc

Um documento R Markdown (*.Rmd) é primeiro compilado para Markdown (*.md) por meio do pacote **knitr** e, em seguida, Markdown é compilado para outros formatos de saída (como LaTeX ou HTML) por meio do Pandoc. Este processo é automatizado pelo pacote **rmarkdown**. Você não precisa instalar o **knitr** ou o **rmarkdown** separadamente, porque eles são os pacotes necessários do **bookdown** e serão instalados automaticamente quando você instalar o **bookdown**. No entanto, o Pandoc não é um pacote R, portanto, não será instalado automaticamente quando você instalar o **bookdown**. Você pode seguir as instruções de instalação na página inicial do Pandoc (<http://pandoc.org>) para instalar o Pandoc, mas se você usa o IDE do RStudio, não precisa instalar o Pandoc separadamente, porque o RStudio inclui uma cópia do Pandoc. O número da versão Pandoc pode ser obtido através de:

```
rmarkdown::pandoc_version()
## [1] '2.17.1.1'
```

Se você achar esta versão muito baixa e houver recursos do Pandoc apenas em uma versão posterior, você poderá instalar a versão posterior do Pandoc e o **rmark down** chamará a versão mais recente em vez de sua versão interna.

A.3 LáTeX

O LáTeX é necessário apenas se você deseja converter seu livro em PDF. Você pode ver <https://www.latex-project.org/get/> para obter mais informações sobre o LáTeX e sua instalação, mas recomendamos que você instale a distribuição LáTeX leve e multiplataforma chamada TinyTeX¹ e baseada no TeX Live. TinyTeX pode ser facilmente instalado através do pacote R **tinytex** (que deve ser instalado automaticamente quando você instala o **bookdown**):

```
tinytex::install_tinytex()
```

Com o TinyTeX, você nunca deve ver mensagens de erro como esta:

```
! Erro LáTeX: Arquivo `titling.sty' não encontrado.
```

```
Digite X para sair ou <RETURN> para continuar,  
ou digite um novo nome. (Extensão padrão: sty)
```

```
Digite o nome do arquivo:
```

```
! Parada de emergência.  
<leia*>
```

```
I.107 ^^M
```

```
pandoc: Erro ao produzir PDF  
Erro: a conversão do documento pandoc falhou com o erro 43  
Execução interrompida
```

O erro acima significa que você usou um pacote que contém `titleling.sty`, mas não foi instalado. Os nomes dos pacotes LáTeX geralmente são os mesmos que os nomes dos arquivos `*.sty`, então neste caso, você pode tentar instalar o pacote de titulação . Se você usa TinyTeX com R Markdown, está faltando o pacote LáTeX

¹ <https://yihui.org/tinytex/>

idades será instalado automaticamente, para que você nunca precise se preocupar com esses problemas.

Distribuições e pacotes LaTeX também são atualizados de tempos em tempos, e você pode considerar atualizá-los especialmente quando tiver problemas com LaTeX. Você pode descobrir a versão da sua distribuição LaTeX por:

```
system('pdflatex --version') ## pdfTeX
3.141592653-2.6-1.40.24 (TeX Live 2022) ## kpathsea versão 6.3.4

## Copyright 2022 Han The Thanh (pdfTeX) et al.
## Não há garantia. A redistribuição deste software é
## coberto pelos termos de direitos autorais do pdfTeX e
## a Licença Pública Geral Menor GNU.
## Para mais informações sobre esses assuntos, veja o arquivo ## chamado COPYING e a
fonte pdfTeX.
## Autor principal do pdfTeX: Han The Thanh (pdfTeX) et al.
## Compilado com libpng 1.6.37; usando libpng 1.6.37 ## Compilado com zlib
1.2.11; usando zlib 1.2.11
## Compilado com xpdf versão 4.03
```

Para atualizar o TinyTeX, você pode executar:

```
tinytex::tlmgr_update()
```

De ano para ano, você também pode precisar atualizar o TinyTeX (caso contrário, você não poderá instalar ou atualizar nenhum pacote LaTeX), caso em que você pode reinstalar o TinyTeX:

```
tinytex::reinstall_tinytex()
```

B

Uso de software

Conforme mencionado no Capítulo 1, este livro não é um guia completo para tricotar ou **rmarkdown**. Neste capítulo, explicamos brevemente alguns conceitos básicos e sintaxe em **knitr** e **rmarkdown**. Se você tem alguma outras perguntas, você pode publicá-las no StackOverflow (<https://stackoverflow.com>) e marque suas perguntas com r, knitr, rmarkdown, e/ou bookdown, o que for apropriado.

B.1 tricô

O pacote **knitr** foi projetado com base na ideia de “Literate Programming” (Knuth, 1984), que permite misturar programas código com texto em um documento de origem. Quando o **knitr** compila um documento, o código do programa (em pedaços de código) será extraído e executado, e a saída do programa será exibida junto com o texto original no documento de saída. Introduzimos a sintaxe básica na Seção 2.3.

R Markdown não é o único formato de origem que o **knitr** suporta. A ideia básica pode ser aplicada a outras linguagens de computação e autoria. Por exemplo, **knitr** também suporta a combinação de R e LaTeX (documentos *.Rnw), e R + HTML (*.Rhtml), etc. Você pode usar outras linguagens de computação com **knitr**, como C++, Python, SQL, e assim por diante. Abaixo está um exemplo simples e você pode ver http://rmarkdown.rstudio.com/authoring_knitr_engines.html para mais.

```
```{python}
x = 'Olá, Mundo Python!'
```

```
print(x.split(' '))
```

Os usuários de Python podem estar familiarizados com IPython ou Jupyter Notebooks (<https://jupyter.org>). Na verdade, o R Markdown também pode ser usado como caderno e tem alguns benefícios adicionais; veja esta postagem do blog para obter mais informações: <https://blog.rstudio.org/2016/10/05/r-notebooks/>.

Se você quiser mostrar um pedaço literal em seu documento, você pode adicionar uma expressão embutida que gera uma string vazia (`r ``) antes do cabeçalho do pedaço, e envolver o pedaço de código em quatro acentos graves,<sup>1</sup> por exemplo,

```
```{r}
# um pedaço de código literal
````
```

Depois que o documento for compilado, a expressão embutida desaparecerá e você verá:

```
``{r}
um pedaço de código literal
````
```

Normalmente, você não precisa chamar funções **knitr** diretamente ao compilar um documento, pois o **rmarkdown** chamará **knitr**. Se você quiser compilar um documento fonte sem convertê-lo em outro for mats, você pode usar a função `knitr::knit()`.

¹Siga a regra de recuo se o trecho de código literal for exibido em outros ambientes, como uma lista: [https://pandoc.org/MANUAL.html#block-content-in-list items](https://pandoc.org/MANUAL.html#block-content-in-list-items)

B.2 R Markdown

Graças ao poder do R e do Pandoc, você pode facilmente fazer cálculos em documentos R Markdown e convertê-los em uma variedade de formatos de saída, incluindo documentos HTML/PDF/Word, slides HTML5/Beamer, painéis e sites, etc. O documento Markdown geralmente consiste nos metadados YAML (opcional) e no corpo do documento.

Introduzimos a sintaxe para escrever vários componentes do corpo do documento no Capítulo 2 e explicamos mais sobre os metadados YAML nesta seção.

Metadados para R Markdown podem ser escritos no início de um documento, começando e terminando com três traços ---, respectivamente.

Os metadados YAML geralmente consistem em pares de valor de tag separados por dois-pontos, por exemplo,

```
---
title: "Um documento R Markdown"
autor: "Yihui Xie"
---
```

Para valores de caracteres, você pode omitir as aspas quando os valores não contiverem caracteres especiais, mas é mais seguro citá-los se for esperado que sejam valores de caracteres.

Além dos caracteres, outro tipo comum de valores são os valores lógicos.

Ambos sim e verdadeiro significam verdadeiro e não/falso significam falso, por exemplo,

```
citações de links: sim
```

Os valores podem ser vetores e há duas maneiras de escrever vetores. As duas maneiras a seguir são equivalentes:

```
saída: ["html_document", "word_document"]
```

```
resultado:  
- "html_document"  
- "documento_palavra"
```

Os valores também podem ser listas de valores. Você só precisa recuar os valores por mais dois espaços, por exemplo,

```
resultado:  
bookdown::gitbook:  
  split_by: "seção"  
  split_bib: não
```

É um erro comum esquecer de recuar os valores. Por exemplo, os seguintes dados

```
resultado:  
html_document:  
toc: sim
```

na verdade significa

```
saída: nulo  
html_document: null  
toc: sim
```

em vez do que você provavelmente teria esperado:

```
resultado:  
html_document:  
toc: sim
```

O formato de saída do R Markdown é especificado no campo de saída dos metadados YAML, e você precisa consultar as páginas de ajuda do R para as opções possíveis, por exemplo, ?rmarkdown::html_document ou ?bookdown::gitbook. Os significados da maioria dos outros campos em YAML podem ser encontrados na documentação do Pan doc.

O pacote **rmarkdown** forneceu essas saídas R Markdown para tapetes:

- beamer_presentation
- documento_contexto
- github_document
- html_document
- ioslides_presentation
- latex_document
- md_document
- odt_document
- pdf_document •
- powerpoint_presentation
- rtf_document
- apresentação_slidy
- documento_palavra

Existem muitos outros formatos de saída possíveis em outros pacotes R, incluindo **os formatos bookdown, tufte, rticles , flexdashboard, revealjs e rmd**, etc.

C

Abaixo está a lista *completa* de perguntas frequentes (FAQ). Sim, há apenas uma pergunta aqui. Pessoalmente eu não gosto de perguntas frequentes. Eles geralmente significam surpresas, e surpresas não são boas para usuários de software.

1. P: O **bookdown** terá os recursos X, Y e Z?

R: A resposta curta é não, mas se você se perguntou três vezes “eu realmente preciso deles” e a resposta ainda for “sim”, sinta-se à vontade para enviar uma solicitação de recurso para <https://github.com/>

Os usuários que solicitam mais recursos geralmente vêm do mundo LaTeX. Se esse for o seu caso, a resposta para essa pergunta é sim, porque o Markdown da Pandoc suporta código LaTeX bruto. Sempre que você sentir que o Markdown não pode fazer o trabalho para você, você sempre tem a opção de aplicar algum código LaTeX bruto em seu documento Markdown. Por exemplo, você pode criar glossários usando o pacote **glossaries**, ou incorporar uma tabela LaTeX complicada, contanto que você conheça a sintaxe do LaTeX. No entanto, lembre-se de que o conteúdo do LaTeX não é portátil. Ele só funcionará para saída LaTeX/PDF e será ignorado em outros tipos de saída. Dependendo da solicitação, podemos portar mais alguns recursos do LaTeX para o **bookdown** no futuro, mas nossa filosofia geral é que o Markdown deve ser mantido o mais simples possível.

A coisa mais desafiadora do mundo não é aprender tecnologias sofisticadas, mas controlar seu próprio coração selvagem.

Bibliografia

Allaire , J. , Xie , Y. , Dervieux , C. , R Foundation , Wickham , H. , Journal of Statistical Software , Vaidyanathan , R. , Association for Computing Machinery , Boettiger , C. , Elsevier , Broman , K. , Mueller, K., Quast, B., Pruij , R., Marwick, B., Wickham, C., Keyes, O., Yu, M., Emaasit, D., Onkelinx, T., Gasparini, A. , Desautels, M.-A., Leutnant, D., MDPI, Taylor e Francis, Öyreden, O., Hance, D., Nüst, D., Uvesten, P., Campitelli, E., Muschelli, J. , Hayes , A. , Camvar , ZN , Ross , N. , Cannoodt , R. , Luguern , D. , Kaplan , DM , Kreutzer , S. , Wang , S. , Hesselberth , J. , and Hyndman , R. . (2022a). *Artigos: Formato de artigo para R Markdown*. Versão do pacote R 0.23.

Allaire, J., Xie, Y., McPherson, J., Luraschi, J., Ushey, K., Atkins, A., Wickham, H., Cheng, J., Chang, W., and Iannone, R. (2022b). *rmarkdown: Documentos Dinâmicos para o pacote R*. R versão 2.14.

Aust, F. (2019). *citr: Suplemento RStudio para inserir citações Markdown*. Versão do pacote R 0.3.2.

Chang, W. (2022). *webshot: Faça capturas de tela de páginas da Web*. pacote R versão 0.5.3.

Cheng, J. (2018). *miniUI: widgets de interface do usuário brilhantes para telas pequenas*. Versão do pacote R 0.1.1.1.

Knuth, DE (1984). Programação alfabetizada. *O Diário do Computador*, 27(2):97-111.

Equipe R Core (2022). *R: Uma Linguagem e Ambiente para Computação Estatística*. R Foundation for Statistical Computing, Viena, Áustria.

Vaidyanathan, R., Xie, Y., Allaire, J., Cheng, J., Sievert, C. e Russell, K. (2021). *htmlwidgets: Widgets HTML para pacote R*. R versão 1.5.4.

Xie, Y. (2015). *Documentos dinâmicos com R e knitr*. Chapman e Hall/CRC, Boca Raton,

Flórida, 2ª edição. ISBN 978-1498716963.

Xie, Y. (2021). *servr: um servidor HTTP simples para servir arquivos estáticos ou dinâmicos Documentos*. Versão do pacote R 0.24.

Xie, Y. (2022a). *bookdown: Livros de autoria e documentos técnicos com R Markdown*.

<https://github.com/rstudio/bookdown>.

Xie, Y. (2022b). *knitr: um pacote de propósito geral para geração de relatórios dinâmicos ração no pacote R.R versão 1.39*.

Xie, Y. e Allaire, J. (2022). *tufte: Estilos de Tufte para RMarkdownDocuments*.

Versão do pacote R 0.12.

Xie, Y., Cheng, J. e Tan, X. (2022). *DT: Um Wrapper das DataTables da Biblioteca*

JavaScript. Versão do pacote R 0.23.

Xie, Y., Dervieux, C. e Riederer, E. (2020). *R Markdown Cookbook*.

Chapman e Hall/CRC, Boca Raton, Flórida. ISBN 9780367563837.

Índice

- _bookdown.yml, 6, 86
- _output.yml, 49, 114
- apêndice, 25
- bookdown.org, 101
- bookdown::publish_book(), 101
- bookdown::render_book(), 6, 91
- bookdown::serve_book(), 94 estilo
- Bootstrap, 68
- Calibre, 76
- citações, 39, 97
- partes de código, 27 referências cruzadas, 15, 19, 29, 33, 38
- CSS, 79
- livro eletrônico, 75
- EPUB, 75
- equação, 15
- figura, 28
- ambiente flutuante, 28, 114 fonte, 114
- GitBook, xi, 50
- GitHub, 98, 105
- HTML, xii, 50, 79
- widget HTML, 44
- índice, 43
- código R embutido, 27
- IPython, 122
- Notebook Jupyter, 122
- tricô, 121
- tricô::include_graphics(), 31
- LaTeX, xii, 2, 73, 80, 113, 119
- Expressão matemática LaTeX, 14, 96
- longtable, 35
- Remarcação, 2, 11
- MathJax, 81
- MOBI, 75
- Pandoc, 11, 118
- Modelo Pandoc, 84 partes, 25 editores, 112
- rmarkdown::render(), 76
- Suplemento RStudio, 96
- RS Studio Connect, 101
- RS Estúdio IDE, 95
- Aplicação brilhante, 47
- tabela,
- teorema 33 , 18
- Travis CI, 106
- Estilo tufado, 72
- YAML, 6, 79, 123