目次

第 1 章

書き方

1.1　読み込んでいるパッケージ

詳しくは “../setting/sotsuron_setting.tex” を見てほしい。特に注意が必要と思われるものを列挙する。

- braket, derivative – ブラケット、微分のパッケージ
- physics2 – physics の代わり
- adjustbox – includegraphics などで max/min width/height を使えるようにする
- otf, pxchfon – フォントを埋め込む。とりあえず ipaex フォントを指定している。好みに合せて変更する。
- biblatex – \addbibresource{../bib/refs.bib} で “../bib/refs.bib” を指定している。

1.2　特有のコマンド

- \sct{ 章の深さ }{ タイトル }[ラベル名]
  セクションなど章のタイトルを label 付きで出力する
  1. 必須 {}: 章の深さ – 0 = chapter, 1 = section, 2 = subsection, 3 = subsubsection
  2. 必須 {}: タイトル – セクションのタイトル
  3. 任意 []: ラベル – ラベル名。デフォルトはタイトルを使用。タイトルに数式を含む場合は必須 (ラベルには数式を入れられない)。
     章の深さに対して、0 = chap: ラベル名, 1 = sec:chapter 名: ラベル名, 2 = s_sec:chapter 名:section 名: ラベル名, 3 = ss_sec:chapter 名:section 名:subsection 名: ラベル名

  タイトルに数式やコマンドを含み、エラーが出る場合は \texorpdfstring{ 数式やコマンド }{ 数式やコマンドを含まないパターン } とする。これは、PDF の目次の部分 (目次の章ではない) に数式やコマンドに対応するものを表示できなかった場合であり、「数式やコマンドを含まないパターン」が代わりに目次に表示される。
  例：章 1.2, 章 1

1.2.1　subsection の例

1.2.2　数式を含む場合 $G = 1$

同一 tex ファイル内で chapter→section→subsectio→subsubsection の順に全て定義しないと、存在しない chpter 名などを参照しようとしてエラーを吐く
なお、zref-xr を用いてファイルをまたいだ ref を可能にしている。

- \fig[表示位置]{ キャプション }[includegraphics のオプション]{ 画像のパス }[ラベル][ラベルの fig: 部分]

画像を表示する。
\fig···[testlabel][figure:]
とラベルに “testlabel”, fig: の部分に “figure:” を指定すると \ref{figure:testlabel} で参照できる。

1. 任意 []: 表示位置 – 図の表示位置。デフォルトは H
2. 必須 {}: キャプション
3. 任意 []: includegraphics のオプション – includegraphics に渡すオプション。この値に関わらず “keepaspectratio, max width=0.9\columnwidth” が指定される。デフォルトは max height=15\baselineskip
4. 必須 {}: 画像のパス
5. 任意 []: ラベル – ラベル名。デフォルトはキャプションを使用
6. 任意 []: ラベルの fig: の部分 – デフォルトは fig:

- \figtwo[表示位置]{ キャプション }[includegraphics のオプション]{ 左の画像のパス }{ 右の画像のパス }[ラベル][ラベルの fig: 部分]
  minipage を用いて画像二つを横並びで表示する。
  includegraphics のオプションに依らず “keepaspectratio, max width=1\columnwidth” が指定される。
  他の引数は \fig の説明を参照。
- \plotcsv[表示位置]{ キャプション }{axis のオプション }{addplot のオプション }{csv のパス }[ラベル][ラベルの fig: 部分]

1. 任意 []: 表示位置 – 図の表示位置。デフォルトは H
2. 必須 {}: キャプション
3. 必須 {}: axis のオプション – axis に渡すオプション。xlabel, ylabel など
4. 必須 {}: addplot のオプション – addplot に渡すオプション。x index, y index など
5. 必須 {}: csv のパス – 括弧とパスの間に空白や改行を入れるとエラーが出る。
6. 任意 []: ラベル – ラベル名。デフォルトはキャプションを使用
7. 任意 []: ラベルの fig: の部分 – デフォルトは fig:

- \TODO{ すること }
  !!TODO!! すること と大きい赤字で表示する
  例 : !!TODO!! すること
- \intii, \intzi, \intiz
  $0, \pm\infty$ などの範囲の積分記号の略記

$$\begin{aligned} \text{\intii} &= \int_{-\infty}^{\infty} \\ \text{\intzi} &= \int_{0}^{\infty} \\ \text{\intiz} &= \int_{-\infty}^{0} \end{aligned} \tag{1.1}$$

- \ctext 文字
  丸 1 などを出力する。
  段落の先頭で用いるとエラーが出るようだ。
  参考 : https://livingdead0812.hatenablog.com/entry/20161005/1475654232
  例 : ①

1.3 数式環境

一行の式

$$e^{i\pi} = -1 \tag{1.2}$$

複数行の式は aligin

$$\nabla \cdot \boldsymbol{B} = 0 \tag{1.3}$$
$$\nabla \cdot \boldsymbol{E} = 0 \tag{1.4}$$

または equation + split

$$\begin{aligned} \nabla \times \boldsymbol{E} &= -\frac{\partial \boldsymbol{B}}{\partial t} \\ \nabla \times \boldsymbol{B} &= \mu_0 \varepsilon_0 \frac{\partial \boldsymbol{E}}{\partial t} \end{aligned} \tag{1.5}$$

これらは式番号の付き方が異なる。aligin では、最後の式に改行 \\ を入れると 1 行分の空白ができてしまう

$$\nabla \cdot \boldsymbol{B} = 0 \tag{1.6}$$
$$\nabla \cdot \boldsymbol{E} = 0 \tag{1.7}$$
$$\tag{1.8}$$

split だと空白は無さそう

$$\begin{aligned} \nabla \times \boldsymbol{E} &= -\frac{\partial \boldsymbol{B}}{\partial t} \\ \nabla \times \boldsymbol{B} &= \mu_0 \varepsilon_0 \frac{\partial \boldsymbol{E}}{\partial t} \end{aligned} \tag{1.9}$$

空白確認用のテキスト

1.4 微積分

$$\frac{\mathrm{d}f}{\mathrm{d}x} = \int \mathrm{d}x\, x \quad (dx \text{ と } x \text{ の間に } \backslash; \text{でスペースを入れる}^{*1}) \tag{1.10}$$
$$\frac{\partial^2 g}{\partial y^2} = \int y\, \mathrm{d}y \tag{1.11}$$
$$\frac{\partial}{\partial z} h(x,y,z) = x + y + z \tag{1.12}$$
$$\mathrm{d}F/\mathrm{d}s = e^{st} \tag{1.13}$$
$$\frac{\partial^3 G}{\partial x\, \partial y\, \partial z} = xyz \tag{1.14}$$
$$\left(\nabla^2 - \mu\varepsilon \frac{\partial^2}{\partial t^2}\right) \boldsymbol{E}(\boldsymbol{r}, t) = 0 \tag{1.15}$$
$$\nabla^2 E = \mu_0 \frac{\partial^2}{\partial t^2} (\varepsilon_0 E + P) \tag{1.16}$$

1.5 図・グラフ、表、リスト

tikz の参考資料 `https://tikz.dev/pgfplots/reference-texdialects`

注意事項

1. here パッケージの H はデフォルトの htbp と一緒に使うとエラーを吐く。画像、表の位置指定では H のみか、htbp の 4 つの組み合わせのみかのどちらか
2. csv ファイルを読み込んでプロットする場合、#が入っているとエラーが出るかもしれない

1.5.1 図・グラフ

参考資料

`https://mirror.aria-on-the-planet.es/CTAN/graphics/pgf/contrib/pgfplots/doc/pgfplots.pdf`

\addplot でグラフをプロットする場合, domain を指定しないと “Missing character: There is no . in font nullfont!” という注意が出る。なお、注意が出ていても問題は無いらしい。

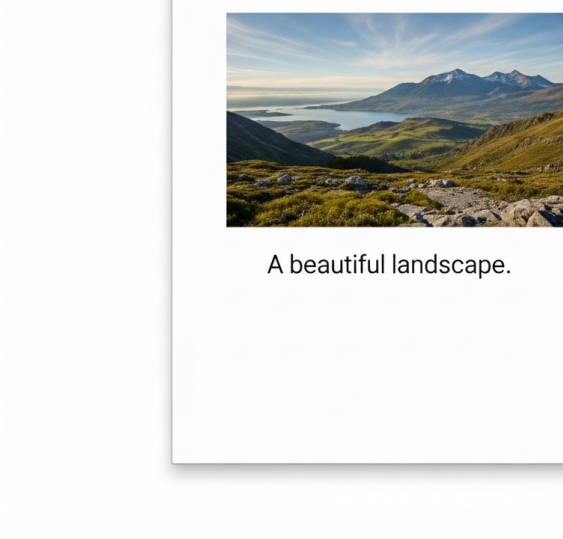

図 1.1: 図の例。Gemini に適当に生成させた画像

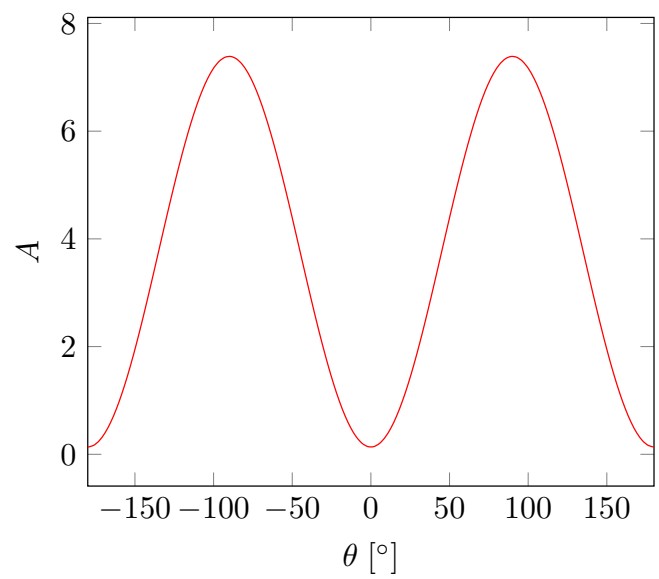

図 1.2: グラフの例

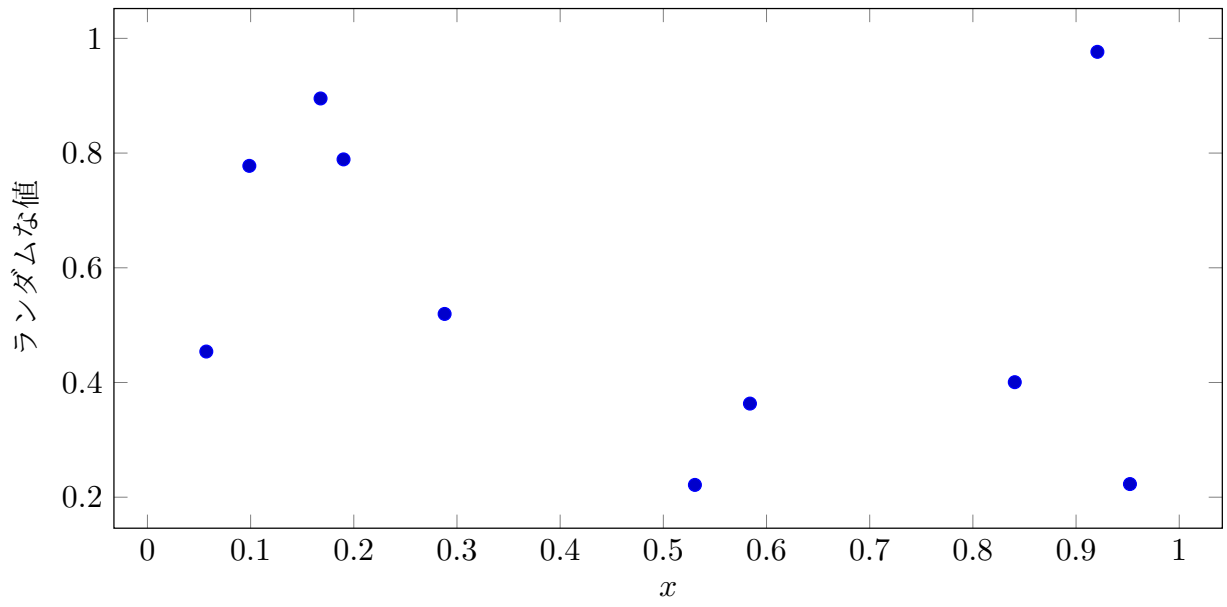

図 1.3: csv のプロット

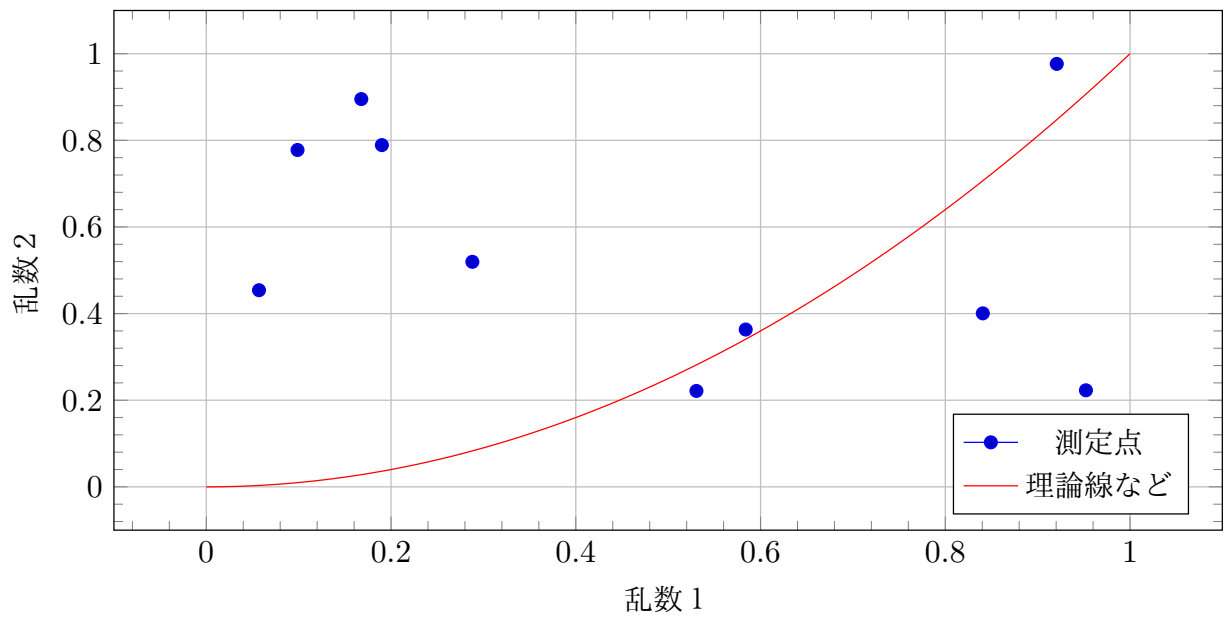

図 1.4: 凡例付き

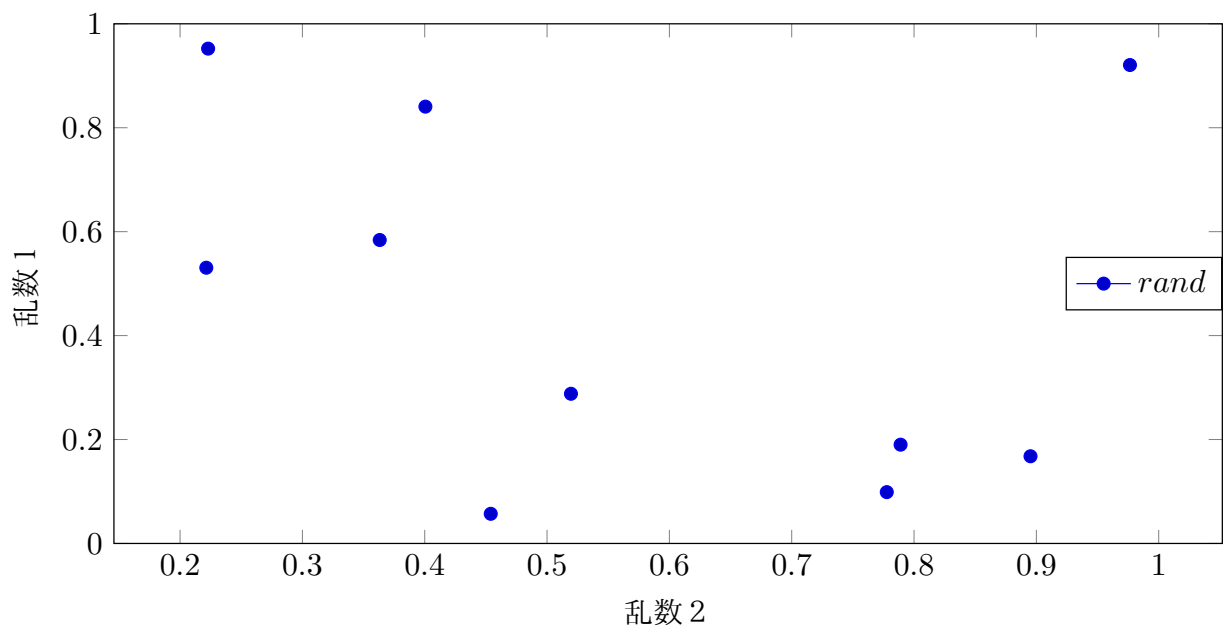

図 1.5: 凡例の位置の調整

1.5.2 表

表 1.1: 表の例

原因	割合
A	4 %
B	5 %
C	2 %
D	3 %
E	1 %
計	15 %

1.5.3 リスト

enumitem パッケージを読み込んでいる前提。leftmargin=*をつけることで、左端が前後の文章と合う。インデントしたい場合は適切な値を入れる。

参照 `https://konoyonohana.blog.fc2.com/blog-entry-58.html`

第 1 章 aaa

aaa について

第 2 章 bbb

bbb について

leftmargin=0pt ではラベル部分は左に飛び出し、テキスト部分の左が左端に合う。

1. aaa
   aaa について
2. bbb
   bbb について

全角一文字分インデントしたい場合は、leftmargin=*, labelindent=1zw とする。

1. aaa
   aaa について
2. bbb
   bbb について

1.6 siunitx

ここにある以外にも \numlist, \numrange, \qtylist などがある `http://www.yamamo10.jp/yamamoto/comp/latex/make_doc/unit/index.php`
単位のみの出力では \si と \unit が、単位と数字では \SI と \qty が使えるが, \si, \SI は古いバージョンとの互換性のために残されているだけで, \unit, \qty を使った方がいいらしい `https://tex.stackexchange.com/questions/603217/should-i-use-qty-or-si-for-siunitx`。ただし physics パッケージと \qty コマンドが被っているため、physics パッケージを使う場合は工夫が必要である。

- 単位付きの数字 100 %
- 誤差と単位付きの数字 3.95(5) dB, 2.23(4) s
- 単位のみ µs
- 数字のみ 2.232×10^4
- 角度 90°, 12.3456°
  度分秒で表記したい場合 12°34′56″ などとする。`https://uec.medit.link/latex/ang.html`
- ミリとメートルの違い。ミリメートルだと間が無く、メートルメートル (そんなものは無いが) では間が空く。m の文字自体には違いは無さそう。因みに平方メートルは m^2 と書く
  ミリメートルと判定されるもの：mm, mm, mm, mm
  メートルメートルと判定されるもの：m m, m m

1.7 脚注、参照

脚注は \footnote を使う[*2]。同じ脚注を複数箇所に付けたい場合は[*3]とラベルを入れておけば \footref で参照できる[*3]。脚注で backref したい場合は, footnotebackref パッケージを読み込めば良い[*4]。

1.8 HyperSnips

1.9 雑多メモ

1. クォーテーション、ダブルクォーテーションは’や”を使うのではなく、バッククォート ‘とクォーテーション’ やバッククォート 2 個 “とクォーテーション 2 個” で表す。記号の向き (?) が変わる。
   ’ クォーテーションで囲う’
   ‘バッククォートとクォーテーション’
   ”ダブルクォーテーションで囲う”、” クォーテーション 2 個で囲う”

[*2] 脚注の例
[*3] 複数箇所につける脚注
[*4] このテンプレートでは読み込んでいない

“バッククォート 2 個とクォーテーション 2 個”

2. LaTeX で使える長さの単位
   \baselineskip：行の高さ, \columnwidth：一行の長さ[*5]
   ”Lengths in LaTeX”. Overleaf
3. \mathrel{}\middle|\mathrel{}：集合の区切り

$$\left\{ x \in \mathbb{C} \;\middle|\; e^{i\gamma x^3} = 1 \right\} \tag{1.17}$$

4. コマンドの引数に特殊な括弧を使うのは latexdiff との相性が悪いかもしれない
   \fig{ キャプション }{ 画像パス }(includegraphics のオプション)
   としていたところ「includegraphics のオプション」部分に \DIFaddbegin や \DIFaddend が入り、コンパイルエラーを吐いた。
   現在は \fig{ キャプション }[includegraphics のオプション]{ 画像パス } とすることで対処している。
   physics2 の ab.braket では \ket|0⟩ などと表記するので章**??**を導入しても動かない可能性がある。
5. \chapter*{ 章の名前 \markboth{ 章の名前 }{}} 章を付けずにタイトル表示する。
   \addcontentsline{toc}{chapter}{ 章の名前 } 章立てせずに目次に追加するおまじない
   \def\chaptername{ 章の名前 } ラベル用に chaptername を定義しておく。
   \markboth はヘッダー左上に表示される文字を指定している。これをしない場合、ヘッダに章番号がついてしまう。
6. \zexternaldocument で読み込んでも「参照先が見つからない」というエラーが出る場合は.tex_intermediates を一度消すとよい。
   また、存在しないファイルを指定しても特にエラーを吐いたりしないため、注意が必要である[*6]。

[*5] 恐らく、2 段組みでは 1 段の行の長さに対応する。\linewidth は 2 段組みでは 2 段合わせた長さに対応する。
[*6] sotsuron.tex でまとめたものではそもそも同一 tex ファイル内に label と ref がある判定になるため、最終的な論文には問題ないはず。

第 2 章

LaTeX の環境構築

環境構築、維持の難易度として 2 段階に分ける

2.1 VSCode で LaTeX を書けるようにする最小構成

1. VSCode, TexLive のインストール ← 調べる
   TexLive のインストールは時間がかかる
2. VSCode を開き、拡張機能をインストールする
   - LaTeX Workshop：LaTeX 書くなら必須
   - Japanese Language Pack for Visual Studio Code：設定画面を日本語にする
3. uplatex でコンパイルするための設定を行う。
   `https://qiita.com/t_kemmochi/items/dd38bbf2b823c770d1ec`
   を参考に「VS Code 内でプレビューする場合の設定」までを行う。
   但し「手順 4. VS Code のその他の設定」で「latex-workshop.latex.recipe.default」は「lastUsed」にする。
   また「latex-workshop.latex.autoClean.run」は「never」にする。
4. 初回コンパイル時は VSCode の画面左の TeX→ LaTeX プロジェクトをビルド → レシピ:latexmk (uplatex) と操作する。
5. 以降は TeX ファイルを保存すると自動でコンパイルされる。
6. VSCode を一度閉じた場合 4 の手順が必要になることがある。

2.1.1 bib ファイルについて

../bib/refs.bib に bib ファイルを置く。
日本語の人名では
“langid = {japanese}”
を追加する。
Zotero を使っている場合、「言語」の欄に “ja” や “japanese” と書いておけば自動で追加される。

2.2 LaTeX を高速にコンパイルできる環境の構成

第 2.1 章の設定から以下の変更を加える。

1. VSCode を開き、拡張機能をインストールする
   以下は必須ではないがオススメのもの

- Path Intellisense：ファイルパスを書くときに補完してくれる。(しない場合は”を先頭に書いてからやるといい)
- HyperSnips：正規表現でスニペットを作れる。自動変換もできる。このフォルダでは h のあとに a-Z0-9 のどれかを打つと \hat{a-Z0-9} となるなど。

2. VSCode の設定を開き、“ユーザー” タブで “Latex-workshop >Latex: Recipes” を調べる。
“settings.json で編集” を選び、setting.json に以下を書く。
/PATH/TO/LATEXMKRC/の部分は各々のファイルの場所に合せて変更する。

```
"latex-workshop.latex.recipes": [
  {
    "name": "latexmk_fast",
    "tools": ["latexmk_fast"]
  },
  ... // 省略
],
"latex-workshop.latex.tools": [
  {
    "name": "latexmk_fast",
    "command": "latexmk",
    "args": [
      "-r", "/PATH/TO/LATEXMKRC/.my_latexmkrc",
      "-g", // 変更が無くてもコンパイルを実行
      "-time", // 実行時間を表示
      "%DOC%"
    ]
  },
  ... // 省略
]
```

3. コンパイルでエラーが出た場合はその TeX ファイルがあるフォルダーの.tex_intermediates を消してから 4 をしてみる

第 3 章

GitHub を用いたバックアップ

GitHub はプログラミング界隈では広く使われているサイト。変更履歴などを残してクラウドにバックアップを取れる。適宜過去の変更地点に戻ることもできるため、“aaa_v1.tex” や “aaa_final(最後) v2.tex” などしなくてよい。VSCode と同じく Microsoft が管理しているので相性がいい。極稀にサイトが落ちるので注意。落ちてもローカルにデータは全てあるので PC 間の同期ができないこと以外に問題はない。以降はバックアップの方法について説明するが、読みにくいので適宜調べてほしい。

3.1 GitHub との連携をする

1. GitHub にアカウントを作成
2. Git をインストールする
3. GitHub で新しい repository を作る
   GitHub で左上の緑の “New” ボタンを押す
4. Repository name はフォルダーと同じイメージ
5. Private を選ぶ
   public(デフォルト) のままだと全世界に公開してしまう
6. ほかはデフォルトで右下の “Create repository”
7. GitHub 上での操作は一旦終わり
8. このフォルダーを PC の適当なフォルダーにコピーする
9. VSCode を開く
10. “フォルダーを開く” でコピーしたフォルダーを開く
11. VSCode の左、上から 3 番目の丸が線で繋がったマークを押す
12. “リポジトリを初期化”
13. “ソース管理” の部分にカーソルをあわせ、右端に表示された “...” を選ぶ
14. “リモート” → “リモートの追加”
15. ログインする
    ここのログイン方法が稀に変化するため、調べる
16. 先ほど作ったリポジトリを選ぶ
17. “変更” の部分にカーソルをあわせ、右端に表示された “+” を選ぶ
18. “✓コミット” の上の “メッセージ” に init などと入力する
    この欄はあとで変更履歴を見るとき、その履歴の名前となる
19. “✓コミット” または Ctrl+Enter でコミット
20. “Branch の発行”
21. GitHub にアクセスしたら更新されているはず

3.2 変更履歴の保存方法

1. VSCode の左、上から 3 番目の丸が線で繋がったマークを押す
2. 他の PC で変更を行った場合、編集する前に “変更の同期” を行い、ローカルのファイルを最新版にする
3. “変更” の部分にカーソルをあわせ、右端に表示された “+” を選ぶ
4. “✓コミット” の上の “メッセージ” に init などと入力する
   この欄はあとで変更履歴を見るとき、その履歴の名前となる
5. “✓コミット” または Ctrl+Enter でコミット
6. “変更の同期”

これを変更履歴を保存したいタイミングでする。多めにしておくと良い。研究室の PC と自分の PC の両方で論文を書く場合、変更履歴が衝突しないようにする必要がある。例えば、自分の PC で編集したが “変更の同期” をする前に、研究室の PC で編集して “変更の同期” を行った場合、Git はクラウドと自分の PC のファイルのどちらが新しいか分からなくなり、面倒なことになる。したがって、編集する PC を変える前に “変更の同期” をしておく必要がある。つまり、研究室から帰るときに “変更の同期” をしておくのがよい。

3.3 タグ

Git タグを使うことで「歴史上の重要なポイントに印をつけることができます」https://git-scm.com/book/ja/v2/Git-%E3%81%AE%E5%9F%BA%E6%9C%AC-%E3%82%BF%E3%82%B0 つまり、特定のコミットに印を付け、素早くアクセスできるようになる。先生に提出した時やそのコメントを反映した時など、ある区切りがついたコミットにつけておくと編集履歴の管理やバックアップの意味で便利である。

タグの作成手順

1. VSCode の左、上から 3 番目の丸が線で繋がったマークを押す
2. ↓ソース管理の右端に・・・があるのでクリック
3. タグからタグの作成
4. コマンドラインで “git push origin タグ名” をする

3.4 pre-commit について

Git には “Git Hooks”, “Git フック” と呼ばれる機能がある。これは様々な Git コマンドが実行される直前または直後に各コマンドに対応した特定のスクリプトを実行する機能である。“pre-commit” は commit コマンドが実行される直前に呼び出される。/pre-commit ファイルの中身は, latexdiff を用いて前回のコミットと今回コミットしようとしている tex ファイルの差分を/diff/diff_ファイル名.tex に書き出し、それを pdf にコンパイルするというものである。この機能を使いたい場合は/pre-commit ファイルを/.git/hooks/以下にコピーする。また、場合によっては “chmod u+x pre-commit” などで実行権限を付与する必要がある。そもそも pdf で差分を見る必要がない、コミットに掛かる時間が非常に伸びる、という点から使用は非推奨とする。

3.5 Branch について

Git には Branch という機能がある。例えば、「章の構造を大きく変えたいけど、戻すかもしれないから今のバージョンも残しておきたい」というときに、ブランチを利用する。デフォルトでは “main” ブランチを使っている。これは”

木”で言うところの “幹” に相等する。ブランチを作成すると、“枝” を作ることができ、“幹” とは別の編集履歴を紡いで行くことができる。大きく変える前に戻したいときはブランチを変更すると戻すことができる。大きく変えた後のものを “幹” にしたい場合はマージするといい。詳しくは各自調べてください。

3.6 Git と GitHub の違い

Git はローカルでバージョン管理ができるもの。変更履歴を “木” として管理している。GitHub は Git の “木” をクラウドにも作る。

コミットはローカルの “木” に変更履歴を追加する。プッシュすると GitHub のクラウド上の “木” とローカルの “木” を同期させる (語弊があるが簡単にした)。

第 4 章

LaTeX 以外の環境構築

4.1 drawio

VSCode に拡張機能の “Draw.io Integration” をインストールする。デフォルトだと黒背景で見ずらいので、ユーザー設定の “Hediet >Vscode-drawio: Theme” を “Kennedy” などにする。また png ファイル全てを drawio で開く設定になっている (.drawio.png 以外の.png も)。“Workbench: Editor Associations” に項目： *.drawio.png, 値： hediet.vscode-drawio を設定する。項目： *.png, 値： hediet.vscode-drawio があれば消す。

拡張子.drawio.png で保存すると, drawio からは編集することができ, LaTeX からは png ファイルとしてそのまま読み込めるので非常に便利である。drawio.png を使う場合、drawio 上で画像サイズが小さいと PDF 上で解像度が悪くなる。drawio のフォントサイズ 40pt~50pt, 線の太さ 4pt 程度に合わせて A4 約 3 ページ以上を目安に、とにかく大きく図を書く。

4.2 Zotero

アドオンの “ZotFile” と “Better BibTex for Zotero” をインストールする。

4.2.1 Better BibTex for Zotero

一つコレクションに引用するものをまとめる。そのコレクションを右クリック、“コレクションをエクスポート”、フォーマット:“Better BibLaTex”、“Keep updated” にチェック、bib ファイルを置く場所を選択する。

bib ファイルの自動更新を切る場合は Zotero の設定から “Better BibTex”、“Open Better BibTex preferences...”、“Automatic export”、自動更新を切る bib ファイルを選択、一番下の “Remove”

第 5 章

latexmk について

latex を pdf に変換する際、ref や cite などのために複数回コマンドを実行する必要がある。例えば, uplatex→ uplatex → biblatex → uplatex → dvipdfmx　など。しかし、この一連の流れを自分で管理するのは困難な上、tex ファイルの中身や編集内容によっては一部を省略することができ、余分に実行すると無駄に時間がかかる。そこで, latexmk を使うと必要に応じて適切なコマンドを実行してくれる。latexmk の設定ファイルは一般に latexmkrc という名前が使われる。latexmkrc は Perl というプログラミング言語で書かれ、$\$latex = `uplatex'$; や $\$latex = `lualatex'$; と書いてコンパイルに使う処理系を uplatex や lualatex と指定することができる。このテンプレートでは “.my_latexmkrc” というファイル名にしている。

5.1　.my_latexmkrc について

元は “ただしい高速 LaTeX 論”. Qiita を参考にした。Windows でも動くように改造してある。変更があった場合のみプリアンブルの読み込みを行うようにすることで、大半の場合でパッケージの読み込みをスキップし、大幅にコンパイル時間を短くすることができる。変更があったかどうかは、プリアンブルを読み込み、空の行や各行の先頭・最後の空白、コメントを無視した部分が前回のコンパイル時と比較して判定している。また、画像も 12 時間キャッシュするようにしてある[*1]。

[*1] 本当にキャッシュされているのかは不明。コンパイルしたときに画像が更新されないような事態は起きていないため放置している。