

Оглавление

8.2. Различные подходы к созданию динамических отчетов	2
8.2.1. Рекомендации по оформлению таблиц.....	2
8.2.2. Оформление таблиц с пакетом <i>flextable</i>	17
8.2.3. Оформление таблиц с пакетом <i>gt</i>	46
8.2.4. Создание отчетов в формате Word.....	75
8.2.5. Создание отчетов в формате PowerPoint	81
8.2.6. Создание отчетов в формате Excel.....	87
8.2.7. Создание отчетов в формате Rmarkdown	90
8.2.8. Фабрика отчетов.....	112

8.2. Различные подходы к созданию динамических отчетов

8.2.1. Рекомендации по оформлению таблиц

Для оформления таблиц в R существует огромное количество пакетов, но в данном издании будут рассмотрены два: *flextable* — как наиболее популярный для создания таблиц для отчетов в форматах MS Office; *gt* — предназначенный для создания таблиц для интерактивных отчетов в формате HTML.

В обоих этих пакетах процесс создания таблицы для отчета выглядит следующим образом: берется набор данных, который мы хотим вывести в отчет; создается объект таблицы, который уже оформляется; в итоге результат выводится в требуемый формат отчета.



Рисунок 8.21. Последовательность действий по созданию табличного файла

Но прежде чем писать код, рассмотрим основные рекомендации о том, как сделать таблицы понятными и информативными для читателя.

Экономист Джон Швебиш в своей статье "Десять рекомендаций по улучшению таблиц", опубликованной в журнале *Journal of Benefit Cost Analysis*¹, сформулировал 10 принципов, которые помогут улучшить представление табличных данных. Хотя не обязательно следовать всем правилам, так как внешний вид таблицы зависит от целей и способа публикации, ознакомиться с представленными рекомендациями будет полезно. Далее рассмотрены принципы оформления таблиц на примере набора данных *cystomegalovirus* (набор данных рассматривался ранее в **разделе 2.6.1.**).

Заголовки таблицы

Важно визуально отделить заголовки столбцов от основной информации в таблице. Это можно сделать, используя жирный шрифт, подчеркивание или линии, чтобы

¹ Источник: <https://www.cambridge.org/core/journals/journal-of-benefit-cost-analysis/article/abs/ten-guidelines-for-better-tables/74C6FD9FEB12038A52A95B9FBCA05A12>

подчеркнуть разницу и сделать таблицу более читаемой. Например простым выделением названий столбцов можно качественно улучшить восприятие таблицы (**рисунки 8.22.-8.23.**).

ID	age	sex	diagnosis.type	TNC.dose	CD34.dose	CD3.dose
1	61	мужской	lymphoid	18.31	2.29	3.21
2	62	мужской	myeloid	4.26	2.04	NA
3	63	женский	myeloid	8.09	6.97	2.19
4	33	женский	myeloid	21.02	6.09	4.87
5	54	женский	myeloid	14.70	2.36	6.55
6	55	мужской	lymphoid	4.29	6.91	2.53

Рисунок 8.22. Пример таблицы без границ

ID	age	sex	diagnosis.type	TNC.dose	CD34.dose	CD3.dose
1	61	мужской	lymphoid	18.31	2.29	3.21
2	62	мужской	myeloid	4.26	2.04	NA
3	63	женский	myeloid	8.09	6.97	2.19
4	33	женский	myeloid	21.02	6.09	4.87
5	54	женский	myeloid	14.70	2.36	6.55
6	55	мужской	lymphoid	4.29	6.91	2.53

Рисунок 8.23. Пример таблицы с выделением названий столбцов

Разделители вместо сетки

Практически никогда не стоит использовать все границы строк, столбцов и ячеек в таблице — это только усложняет восприятие. Гораздо эффективнее выделять границами или отступами важные смысловые блоки таблицы. Это можно делать как по вертикали, так и по горизонтали. Для примера ниже представлена таблица, в которой все ячейки выделены границами (**рисунок 8.24.**).

	time.to.cmv	time.to.agvh	time.to.cgvhd	TNC.dose	CD34.dose	CD3.dose
Hodgkin lymphoma	28.84	12.99	11.01	17.42	4.22	4.91
acute lymphoblastic leukemia	4.37	2.79	4.37	14.70	2.36	6.55
acute myeloid leukemia	4.67	6.89	5.48	10.32	4.96	4.66
aplastic anemia	3.88	2.60	3.88	5.25	4.45	2.31
chronic lymphocytic leukemia	10.44	30.12	7.81	9.13	4.97	2.77
chronic myeloid leukemia	25.93	42.14	20.48	10.77	7.40	5.21
Average	13.02	16.26	8.84	11.26	4.73	4.40

Рисунок 8.24. Пример таблицы с выделением всех ячеек

Ориентироваться в таблице, представленной на **рисунке 8.24.** достаточно затруднительно. Для улучшения визуального восприятия необходимо выделить строку итогов и группы переменных, связанные со временем и дозировкой. Дополнительно можно добавить заголовок для групп столбцов (**рисунок 8.25.**).

diagnosis	time to			dose		
	cmv	agvh	cgvhd	TNC	CD34	CD3
Hodgkin lymphoma	28.84	12.99	11.01	17.42	4.22	4.91
acute lymphoblastic leukemia	4.37	2.79	4.37	14.70	2.36	6.55
acute myeloid leukemia	4.67	6.89	5.48	10.32	4.96	4.66
aplastic anemia	3.88	2.60	3.88	5.25	4.45	2.31
chronic lymphocytic leukemia	10.44	30.12	7.81	9.13	4.97	2.77
chronic myeloid leukemia	25.93	42.14	20.48	10.77	7.40	5.21
Average	13.02	16.26	8.84	11.26	4.73	4.40

Рисунок 8.25. Пример таблицы с группировкой столбцов

Выравнивание чисел по правому краю

Для простоты восприятия и сравнения чисел в таблице рекомендуется выравнивать их по правому краю. Это особенно важно для дробных чисел: их нужно выравнивать по десятичной точке. Иногда для достижения идеального выравнивания может потребоваться добавить нули к дробной части или округлить числа. Такой подход добавит таблице структуированности и сделает ее более удобной для чтения. Далее представлен пример сравнения таблицы с различными направлениями выравнивания числовых столбцов. В первой таблице на **рисунке 8.26.** использованы числа без визуального улучшения. В таблице на **рисунке 8.27.** все числа приведены к одинаковому количеству цифр в дробной части.

diagnosis	левый	центр	правый
Hodgkin lymphoma	28.8367	12.99	11.0067
acute lymphoblastic leukemia	4.37	2.79	4.37
acute myeloid leukemia	4.6733	6.8933	5.4817
aplastic anemia	3.88	2.6	3.88
chronic lymphocytic leukemia	10.438	30.116	7.808
chronic myeloid leukemia	25.9325	42.1425	20.485

Рисунок 8.26. Пример таблицы без выравнивания чисел

diagnosis	левый	центр	правый
Hodgkin lymphoma	28.84	12.99	11.01
acute lymphoblastic leukemia	4.37	2.79	4.37
acute myeloid leukemia	4.67	6.89	5.48
aplastic anemia	3.88	2.60	3.88
chronic lymphocytic leukemia	10.44	30.12	7.81
chronic myeloid leukemia	25.93	42.14	20.48

Рисунок 8.27. Пример таблицы с выравниванием чисел

Выравнивание чисел по правому краю действительно упрощает сравнение, особенно для определения самого большого числа. Но есть один нюанс: это работает только для шрифтов, где ширина цифр одинакова. В шрифтах Segoe UI и Arial десятичная точка располагается ровно на одной линии. Это позволяет легко сравнивать дробные числа, даже если они имеют разное количество цифр после запятой. Однако, в шрифтах Cabin и Georgia десятичная точка может "гулять" по горизонтали, из-за чего выравнивание по правому краю становится неэффективным (**рисунок 8.28.**). Следует отметить, что при

выборе шрифта для таблицы с числами необходимо, чтобы он обеспечивал четкое выравнивание по правому краю.

diagnosis	Segoe UI	Arial	Cabin	Georgia
Hodgkin lymphoma	12.99	12.99	12.99	12.99
acute lymphoblastic leukemia	2.79	2.79	2.79	2.79
acute myeloid leukemia	6.89	6.89	6.89	6.89
aplastic anemia	2.60	2.60	2.60	2.60
chronic lymphocytic leukemia	30.12	30.12	30.12	30.12
chronic myeloid leukemia	42.14	42.14	42.14	42.14
Average	16.26	16.26	16.26	16.26

Рисунок 8.28. Выравнивание чисел в таблице с разными шрифтами

Безусловно, большинство пакетов для оформления таблиц использует корректные шрифты по умолчанию, но при применении собственных стилей нужно учитывать представленные особенности выравнивания.

Выравнивание текста по левому краю

Для текста в таблице рекомендуется устанавливать выравнивание по левому краю. Это позволяет легко следить за текстом по вертикали, независимо от его длины, и создает четкую границу между столбцами (**рисунок 8.29.**).

левый	центр	правый
Hodgkin lymphoma	Hodgkin lymphoma	Hodgkin lymphoma
acute lymphoblastic leukemia	acute lymphoblastic leukemia	acute lymphoblastic leukemia
acute myeloid leukemia	acute myeloid leukemia	acute myeloid leukemia
aplastic anemia	aplastic anemia	aplastic anemia
chronic lymphocytic leukemia	chronic lymphocytic leukemia	chronic lymphocytic leukemia
chronic myeloid leukemia	chronic myeloid leukemia	chronic myeloid leukemia

Рисунок 8.29. Пример таблицы с длинным текстом с различными вариантами выравнивания

Допускается выравнивание текстовых столбцов по центру, если длина текста небольшая и вариативность значений маленькая (например, если значения столбца “да/нет”). Такой подход может улучшить восприятие информации (**рисунок 8.30.**)

левый	центр	правый
acute myeloid leukemia	да	61
non-Hodgkin lymphoma	нет	62
non-Hodgkin lymphoma	нет	63
Hodgkin lymphoma	нет	33
acute lymphoblastic leukemia	нет	54
myelofibrosis	да	55

Рисунок 8.30. Пример таблицы с выравниванием текста оптимальным образом

Адекватный уровень точности

Далеко не всегда нужна точность в числах до четырех или пяти знаков после запятой — обычно достаточно двух знаков. Нужно найти баланс между информационной ценностью и аккуратностью таблицы. Например, в таблице на **рисунке 8.31.** представлены средние значения дозировок с различным вариантом округления. Исходные данные были с точностью в два знака после запятой, поэтому имеет смысл сохранить эту точность.

diagnosis	Много	Мало	В самый раз
Hodgkin lymphoma	17.42333	17	17.42
acute lymphoblastic leukemia	14.70000	15	14.70
acute myeloid leukemia	10.31833	10	10.32
aplastic anemia	5.25000	5	5.25
chronic lymphocytic leukemia	9.12800	9	9.13
chronic myeloid leukemia	10.76500	11	10.77

Рисунок 8.31. Пример таблицы с различной точностью округления

Направление внимания с помощью отступов

Использование пробелов и отступов в таблице позволяет акцентировать внимание читателя на нужных значениях и упростить понимание данных. Схожие по смыслу данные логичнее размещать ближе друг к другу. Если требуется сравнивать показатели, то также рекомендуется разместить их рядом. Скученный текст также может негативно сказаться на понимании. На **рисунке 8.32.** представлена таблица с неоптимальными отступами: информация кажется скученной и ее восприятие затруднено.

diagnosis	time to			dose		
	cmv	agvh	cgvhd	TNC	CD34	CD3
Hodgkin lymphoma	28.84	12.99	11.01	17.42	4.22	4.91
acute lymphoblastic leukemia	4.37	2.79	4.37	14.70	2.36	6.55
acute myeloid leukemia	4.67	6.89	5.48	10.32	4.96	4.66
aplastic anemia	3.88	2.60	3.88	5.25	4.45	2.31
chronic lymphocytic leukemia	10.44	30.12	7.81	9.13	4.97	2.77
chronic myeloid leukemia	25.93	42.14	20.48	10.77	7.40	5.21
Average	13.02	16.26	8.84	11.26	4.73	4.40

Рисунок 8.32. Пример таблицы с неоптимальными отступами

Если предполагается, что данные в таблице нужно читать сверху вниз, то рекомендуется увеличить отступы между столбцами и уменьшить отступы между строками. Это сделает сравнение значений по вертикали более удобным.

diagnosis	time to			dose		
	cmv	agvh	cgvhd	TNC	CD34	CD3
Hodgkin lymphoma	28.84	12.99	11.01	17.42	4.22	4.91
acute lymphoblastic leukemia	4.37	2.79	4.37	14.70	2.36	6.55
acute myeloid leukemia	4.67	6.89	5.48	10.32	4.96	4.66
aplastic anemia	3.88	2.60	3.88	5.25	4.45	2.31
chronic lymphocytic leukemia	10.44	30.12	7.81	9.13	4.97	2.77
chronic myeloid leukemia	25.93	42.14	20.48	10.77	7.40	5.21
Average	13.02	16.26	8.84	11.26	4.73	4.40

Рисунок 8.33. Пример таблицы с оптимальными отступами для чтения по вертикали

Если наиболее частый сценарий чтения таблицы — по горизонтали, то рекомендуется уменьшить отступы между столбцами и увеличить отступы между строками. Такой подход помогает глазу плавно скользить по строке, не перескакивая на соседнюю.

diagnosis	time to			dose		
	cmv	agvh	cgvhd	TNC	CD34	CD3
Hodgkin lymphoma	28.84	12.99	11.01	17.42	4.22	4.91
acute lymphoblastic leukemia	4.37	2.79	4.37	14.70	2.36	6.55
acute myeloid leukemia	4.67	6.89	5.48	10.32	4.96	4.66
aplastic anemia	3.88	2.60	3.88	5.25	4.45	2.31
chronic lymphocytic leukemia	10.44	30.12	7.81	9.13	4.97	2.77
chronic myeloid leukemia	25.93	42.14	20.48	10.77	7.40	5.21
Average	13.02	16.26	8.84	11.26	4.73	4.40

Рисунок 8.34. Пример таблицы с оптимальными отступами для чтения по горизонтали
Для экономии места в таблице можно использовать чередование фона строк. Это позволит максимально уменьшить отступы между строками, сохраняя при этом читаемость данных. Глаз легко будет следить за строками благодаря контрасту цветов, создаваемому чередующейся заливкой (**рисунок 8.35.**)

diagnosis	time to			dose		
	cmv	agvh	cgvhd	TNC	CD34	CD3
Hodgkin lymphoma	28.84	12.99	11.01	17.42	4.22	4.91
acute lymphoblastic leukemia	4.37	2.79	4.37	14.70	2.36	6.55
acute myeloid leukemia	4.67	6.89	5.48	10.32	4.96	4.66
aplastic anemia	3.88	2.60	3.88	5.25	4.45	2.31
chronic lymphocytic leukemia	10.44	30.12	7.81	9.13	4.97	2.77
chronic myeloid leukemia	25.93	42.14	20.48	10.77	7.40	5.21
Average	13.02	16.26	8.84	11.26	4.73	4.40

Рисунок 8.35. Пример таблицы с чередованием цвета строк

Единицы измерения

Предполагается, что в рамках столбца все числовые значения записаны в одинаковых единицах измерения, поэтому нет смысла дублировать их в каждой ячейке. Если к символам валют, процентов, градусов можно отнести снисходительно, особенно, если они находятся только в одном столбце, то во всех остальных случаях такой подход создает излишнюю визуальную нагрузку и мешает адекватному восприятию таблицы. В таких случаях рекомендуется выносить единицы измерения в заголовки столбцов (рисунки 8.36.-8.37.).

diagnosis	TNC	CD34	CD3
Hodgkin lymphoma	17.42 x10 ⁸ /kg	4.22 x10 ⁶ /kg	4.91 x10 ⁸ /kg
acute lymphoblastic leukemia	14.70 x10 ⁸ /kg	2.36 x10 ⁶ /kg	6.55 x10 ⁸ /kg
acute myeloid leukemia	10.32 x10 ⁸ /kg	4.96 x10 ⁶ /kg	4.66 x10 ⁸ /kg
aplastic anemia	5.25 x10 ⁸ /kg	4.45 x10 ⁶ /kg	2.31 x10 ⁸ /kg
chronic lymphocytic leukemia	9.13 x10 ⁸ /kg	4.97 x10 ⁶ /kg	2.77 x10 ⁸ /kg
chronic myeloid leukemia	10.77 x10 ⁸ /kg	7.40 x10 ⁶ /kg	5.21 x10 ⁸ /kg
Average	11.26 x10⁸/kg	4.73 x10⁶/kg	4.40 x10⁸/kg

Рисунок 8.36. Пример таблицы с избыточным использованием единиц измерения

diagnosis	TNC, x10⁸/kg	CD34, x10⁶/kg	CD3, x10⁸/kg
Hodgkin lymphoma	17.42	4.22	4.91
acute lymphoblastic leukemia	14.70	2.36	6.55
acute myeloid leukemia	10.32	4.96	4.66
aplastic anemia	5.25	4.45	2.31
chronic lymphocytic leukemia	9.13	4.97	2.77
chronic myeloid leukemia	10.77	7.40	5.21
Average	11.26	4.73	4.40

Рисунок 8.37. Пример таблицы с единицами измерения в заголовках столбцов

На **рисунке 8.37.** ввиду множества аббревиатур восприятие таблицы остается затруднительным. Кроме того, для двух столбцов единицы измерения повторяются. В этом случае рекомендуется разместить расшифровку наименований столбцов в заметках под таблицей. Это также позволит добавить дополнительное описание к столбцам.

diagnosis	TNC¹	CD34²	CD3³
Hodgkin lymphoma	17.42	4.22	4.91
acute lymphoblastic leukemia	14.70	2.36	6.55
acute myeloid leukemia	10.32	4.96	4.66
aplastic anemia	5.25	4.45	2.31
chronic lymphocytic leukemia	9.13	4.97	2.77
chronic myeloid leukemia	10.77	7.40	5.21
Average	11.26	4.73	4.40

¹ Total nucleated cell dose, x10⁸/kg
² CD34+ cell dose, x10⁶/kg
³ CD3+ cell dose, x10⁸/kg

Рисунок 8.38. Пример таблицы с оптимальным описанием заголовков столбцов

Выбросы и важные значения

При работе с большими таблицами бывает полезно выделить выбросы значений цветом или формой. Это поможет быстро привлечь внимание к аномалиям и сразу понять, какие данные требуют дополнительного анализа.

diagnosis	White		African American	
	Male	Female	Female	Male
acute myeloid leukemia	57	58	-	61
non-Hodgkin lymphoma	55	55	-	62
myelodysplastic syndrome	55	57	-	-
chronic lymphocytic leukemia	56	51	48	-
multiple myelomas	43	49	40	-
chronic myeloid leukemia	57	46	62	-
myelofibrosis	56	54	-	-
renal cell carcinoma	48	38	-	-
Hodgkin lymphoma	49	33	-	-
acute lymphoblastic leukemia	-	54	-	-
aplastic anemia	36	-	-	-
congenital anemia	42	-	-	-
myeloproliferative disorder	50	-	-	-

Рисунок 8.39. Пример таблицы без цветового выделения выбросов

diagnosis	White		African American	
	Male	Female	Female	Male
acute myeloid leukemia	57	58	-	61
non-Hodgkin lymphoma	55	55	-	62
myelodysplastic syndrome	55	57	-	-
chronic lymphocytic leukemia	56	51	48	-
multiple myelomas	43	49	40	-
chronic myeloid leukemia	57	46	62	-
myelofibrosis	56	54	-	-
renal cell carcinoma	48	38	-	-
Hodgkin lymphoma	49	33	-	-
acute lymphoblastic leukemia	-	54	-	-
aplastic anemia	36	-	-	-
congenital anemia	42	-	-	-
myeloproliferative disorder	50	-	-	-

Рисунок 8.40. Пример таблицы с цветовым выделением выбросов

diagnosis	White		African American	
	Male	Female	Female	Male
acute myeloid leukemia	57	58	-	61
non-Hodgkin lymphoma	55	55	-	62
myelodysplastic syndrome	55	57	-	-
chronic lymphocytic leukemia	56	51	48	-
multiple myelomas	43	49	40	-
chronic myeloid leukemia	57	46	62	-
myelofibrosis	56	54	-	-
renal cell carcinoma	48	38	-	-
Hodgkin lymphoma	49	33	-	-
acute lymphoblastic leukemia	-	54	-	-
aplastic anemia	36	-	-	-
congenital anemia	42	-	-	-
myeloproliferative disorder	50	-	-	-

Рисунок 8.41. Пример таблицы с более интенсивным цветовым выделением выбросов

Таблица на **рисунке 8.40.** использует цвет шрифта, чтобы подчеркнуть важные моменты: возраст менее 40 лет выделен красным, а возраст более 60 лет — зеленым. Это помогает быстро найти нужную информацию и сосредоточиться на интересующих нас данных. В таблице на **рисунке 8.41.** для выделения информации используется заливка, заставляя читателя сосредоточиться только на окрашенных значениях. При использовании цвета в таблице необходимо соблюдать логику и сохранять ее во всех столбцах. Например, если зеленым цветом выделяются самые большие значения, то так нужно делать во всех столбцах, чтобы не сбивать читателя.

Группировка данных

Если в таблице присутствует большое количество повторяющихся значений, рекомендуется объединить их, чтобы уменьшить визуальный шум. Одним из способов решения данной задачи является группировка строк (**рисунки 8.42.-8.43.**). Другим способом является удаление избыточно повторяющихся значений (**рисунок 8.44.**). В обоих способах главным является визуальная группировка данных по смыслу.

diagnosis.type	diagnosis	n	time
lymphoid	acute myeloid leukemia	12	14.12
lymphoid	chronic myeloid leukemia	4	9.42
lymphoid	myelodysplastic syndrome	9	9.28
lymphoid	myelofibrosis	4	6.81
lymphoid	myeloproliferative disorder	1	173.83
myeloid	Hodgkin lymphoma	3	34.89
myeloid	acute lymphoblastic leukemia	1	11.40
myeloid	chronic lymphocytic leukemia	5	73.49
myeloid	multiple myelomas	7	21.16
myeloid	non-Hodgkin lymphoma	12	65.29

Рисунок 8.42. Пример таблицы с повторяющимися значениями столбцов

diagnosis	n	time
lymphoid		
acute myeloid leukemia	12	14.12
chronic myeloid leukemia	4	9.42
myelodysplastic syndrome	9	9.28
myelofibrosis	4	6.81
myeloproliferative disorder	1	173.83
myeloid		
Hodgkin lymphoma	3	34.89
acute lymphoblastic leukemia	1	11.40
chronic lymphocytic leukemia	5	73.49
multiple myelomas	7	21.16
non-Hodgkin lymphoma	12	65.29

Рисунок 8.43. Пример таблицы с сгруппированными значениями для исключения повторов

diagnosis.type	diagnosis	n	time
lymphoid	acute myeloid leukemia	12	14.12
	chronic myeloid leukemia	4	9.42
	myelodysplastic syndrome	9	9.28
	myelofibrosis	4	6.81
	myeloproliferative disorder	1	173.83
myeloid	Hodgkin lymphoma	3	34.89
	acute lymphoblastic leukemia	1	11.40
	chronic lymphocytic leukemia	5	73.49
	multiple myelomas	7	21.16
	non-Hodgkin lymphoma	12	65.29

Рисунок 8.44. Пример таблицы с удаленными избыточными значениями

Визуализация

Небольшие визуализации, добавленные к таблице могут кардинально изменить ее восприятие. Они делают таблицу более наглядной и позволяют проще выделять тренды и паттерны, на которые необходимо обратить внимание читателя. Например, можно добавить график boxplot (см. раздел 4.3.), чтобы дополнительно проиллюстрировать распределение значений (**рисунок 8.45.**).

diagnosis	n	min	avg	max	boxplot
acute myeloid leukemia	12	45	58	67	
non-Hodgkin lymphoma	12	41	56	63	
myelodysplastic syndrome	9	35	56	64	
multiple myelomas	7	34	44	60	
chronic lymphocytic leukemia	5	48	54	59	
chronic myeloid leukemia	4	39	52	62	

Рисунок 8.45. Пример таблицы с графиком boxplot

Для иллюстрации большого количества однотипных значений можно добавить окрашивание heatmap (см. раздел 4.3.) в зависимости от значения, чтобы подчеркнуть диапазон разброса (рисунок 8.46.).

diagnosis	trnsp	cmv	agvhhd	cgvhd
acute myeloid leukemia	14.12	4.67	6.89	5.48
non-Hodgkin lymphoma	65.29	16.83	21.82	11.78
myelodysplastic syndrome	9.28	10.02	10.63	14.10
multiple myelomas	21.16	12.99	7.55	15.08
chronic lymphocytic leukemia	73.49	10.44	30.12	7.81
chronic myeloid leukemia	9.42	25.93	42.14	20.48

Рисунок 8.46. Пример таблицы с окрашиванием heatmap

В данном разделе были представлены основные подходы для улучшения визуального восприятия таблиц. Далее будут рассмотрены примеры кода на языке R для реализации данных подходов на практике.

8.2.2. Оформление таблиц с пакетом *flextable*

Одним из самых популярных пакетов для оформления таблиц в R является пакет *flextable*. Таблицы, созданные с его помощью можно использовать при оформлении

отчетов, а также сохранять отдельно во всех популярных форматах: DOCX, PDF, HTML, PNG. Обычно данный пакет используется для формирования отчетов с пакетом *officer*.

Логика работы с пакетом похожа на конвейер, который обсуждался ранее при работе с пакетом *dplyr*: за основу берется *data.frame* с данными, на основе него создается объект *flextable*, а затем с этим объектом последовательно производятся операции, меняющие его оформление. Сами операции можно применять последовательно друг за другом и объединять в цепочку с помощью оператора `%>%`. Для примера будет оформлена таблица, которая была создана в начале [раздела 2.6.3](#).

```
data
# A tibble: 7 × 5
  diagnosis sex      n percent_diag percent_overall
  <chr>     <chr>   <int>        <dbl>        <dbl>
1 Lymphoid Женский    15        50          23.4
2 Lymphoid Мужской    15        50          23.4
3 myeloid Женский    13       46.4         20.3
4 myeloid Мужской    15       53.6         23.4
5 NA        Женский     2        33.3         3.12
6 NA        Мужской     4        66.7         6.25
7 Всего      Всего      64        NA           100
```

Чтобы создать объект *flextable* достаточно передать *data.frame* в функцию *flextable()*. Результат сразу отобразится в RStudio в окне Viewer (**рисунок 8.47**).

```
ft <- flextable(data)
ft
```

diagnosis	sex	n	percent_diag	percent_overall
lymphoid	Женский	15	50.00000	23.4375
lymphoid	Мужской	15	50.00000	23.4375
myeloid	Женский	13	46.42857	20.3125
myeloid	Мужской	15	53.57143	23.4375
	Женский	2	33.33333	3.1250
	Мужской	4	66.66667	6.2500
Всего	Всего	64		100.0000

Рисунок 8.47. Пример таблицы *flextable*

Следует отметить, что не обязательно выводить в таблицу все столбцы из *data.frame* — можно указать конкретные столбцы, которые необходимо использовать при визуализации. Дополнительно можно определить ширину столбцов и тему оформления таблицы (**рисунок 8.48**).

```
ft <- flextable(data, col_keys = c("diagnosis", "sex", "n", "percent_overall"))
ft
```

diagnosis	sex	n	percent_overall
lymphoid	Женский	15	23.4375
lymphoid	Мужской	15	23.4375
myeloid	Женский	13	20.3125
myeloid	Мужской	15	23.4375
	Женский	2	3.1250
	Мужской	4	6.2500
Всего	Всего	64	100.0000

Рисунок 8.48. Пример таблицы *flextable* с избранными столбцами

По умолчанию объект *flextable* состоит из трех частей (**рисунок 8.49.**):

1. Заголовок (header) - находится непосредственно над данными. По умолчанию включает только заголовки столбцов (colnames).
2. Тело (body) - данные таблицы (строки *data* из *data.frame*).
3. Футер (footer) - находится под данными. По умолчанию отсутствует (empty).

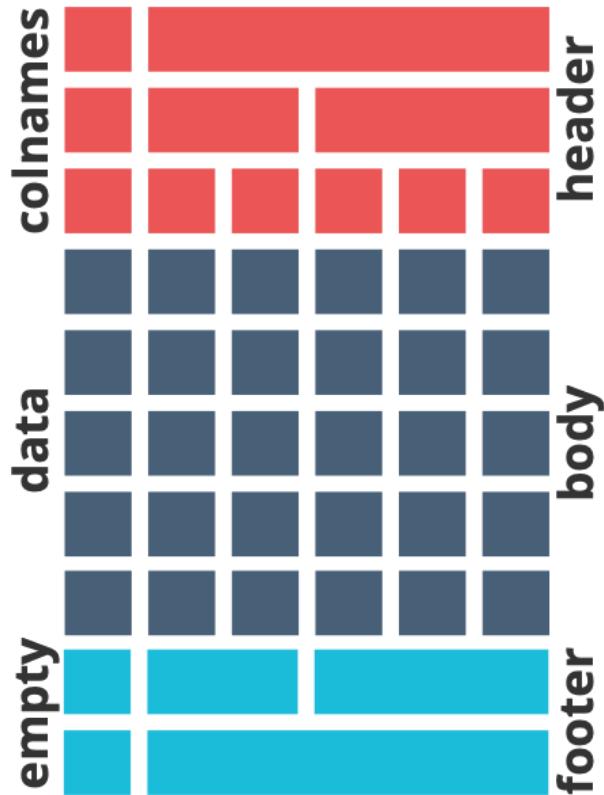


Рисунок 8.49. Общая структура таблицы *flextable*

К каждой из частей таблицы *flextable* можно добавлять дополнительные строки с помощью функций, при этом следует учитывать порядок выполнения команд. С помощью параметра *colwidths* указывается количество столбцов на которые распространяется элемент строки (рисунок 8.50.).

```
ft <- add_header_row(ft, values = c("Категория", "Значение"), colwidths = c(2, 2))
ft <- add_header_row(ft, values = "Первая строка заголовка", colwidths = 4)
ft <- add_footer_lines(ft, "Примечание к таблице")
ft
```

Первая строка заголовка			
Категория		Значение	
diagnosis	sex	n	percent_overall
lymphoid	Женский	15	23.4375
lymphoid	Мужской	15	23.4375
myeloid	Женский	13	20.3125
myeloid	Мужской	15	23.4375
	Женский	2	3.1250
	Мужской	4	6.2500
Всего	Всего	64	100.0000

Примечание к таблице

Рисунок 8.50. Таблица *flextable* с двумя добавленными заголовками

При добавлении заголовка нужно указать вектор подписей и количество столбцов, на которые он распространяется. Следует отметить, что заголовки добавляются последовательно снизу вверх, а подписи в футере сверху вниз. Гораздо удобнее работать с таблицами с помощью оператора пайп `%>%`. С использованием данного оператора команды можно переписать следующим образом.

```
ft <- data %>% flextable(col_keys = c("diagnosis", "sex", "n", "percent_overall"))
%>%
  add_header_row(values = c("Категория", "Значение"), colwidths = c(2, 2)) %>%
  add_header_row(values = "Первая строка заголовка", colwidths = 4) %>%
  add_footer_lines("Примечание к таблице")
```

После того как была разобрана общая логика работы с таблицами *flextable* следует оформить таблицу в соответствии с представленными в разделе 8.2.1. рекомендациями. Можно отметить, что создаваемые по умолчанию таблицы уже соответствуют некоторым рекомендациям: заголовки столбцов отделяются визуально, текст выравнивается по левому краю, цифры выравниваются по правому краю и т.д. Это во многом достигается за счет использования встроенных тем оформления, которые уже содержат настроенные параметры для отображения таблиц. В пакете *flextable* присутствует большое количество тем оформления таблиц. По умолчанию используется тема *booktabs*, также подходящими для печати темами являются: *apa*, *box*, *alafoli*, *vanilla*, *zebra*.

```
ft %>% theme_box()
```

Первая строка заголовка			
Категория		Значение	
diagnosis	sex	n	percent_overall
lymphoid	Женский	15	23.4375
lymphoid	Мужской	15	23.4375
myeloid	Женский	13	20.3125
myeloid	Мужской	15	23.4375
	Женский	2	3.1250
	Мужской	4	6.2500
Всего	Всего	64	100.0000

Примечание к таблице

Рисунок 8.51. Таблица *flextable* с темой *box* (функция *theme_box()*)

```
ft %>% theme_vanilla()
```

Первая строка заголовка			
Категория		Значение	
diagnosis	sex	n	percent_overall
lymphoid	Женский	15	23.4375
lymphoid	Мужской	15	23.4375
myeloid	Женский	13	20.3125
myeloid	Мужской	15	23.4375
	Женский	2	3.1250
	Мужской	4	6.2500
Всего	Всего	64	100.0000

Примечание к таблице

Рисунок 8.52. Таблица *flextable* с темой *vanilla* (функция *theme_vanilla()*)

```
ft %>% theme_zebra()
```

Первая строка заголовка			
Категория		Значение	
diagnosis	sex	n	percent_overall
lymphoid	Женский	15	23.4375
lymphoid	Мужской	15	23.4375
myeloid	Женский	13	20.3125
myeloid	Мужской	15	23.4375
	Женский	2	3.1250
	Мужской	4	6.2500
Всего	Всего	64	100.0000

Примечание к таблице

Рисунок 8.53. Таблица *flextable* с темой *zebra* (функция *theme_zebra()*)

Такие темы удобно использовать как основу и настраивать конечное оформление с помощью дополнительных функций. Их можно применять с помощью соответствующих функций и целиком менять визуальное оформление таблицы. Далее в качестве примера будет оформляться таблица *data* со стандартной темой *booktabs*.

```
ft <- data %>% flextable() %>% theme_booktabs()
ft
```

diagnosis	sex	n	percent_diag	percent_overall
lymphoid	Женский	15	50.00000	23.4375
lymphoid	Мужской	15	50.00000	23.4375
myeloid	Женский	13	46.42857	20.3125
myeloid	Мужской	15	53.57143	23.4375
	Женский	2	33.33333	3.1250
	Мужской	4	66.66667	6.2500
Всего	Всего	64		100.0000

Рисунок 8.54. Таблица *flextable* с темой *booktabs*(функция *theme_booktabs()*)

Заголовки и примечания

Согласно первой, рассмотренной ранее, рекомендации необходимо выделить заголовки столбцов. По умолчанию они уже выделяются в рамках стандартной темы. Далее представлен пример переименования названий столбцов, поскольку сейчас в качестве заголовков используются названия переменных.

```
ft %>% set_header_labels(diagnosis = "Диагноз",
                           sex = "Пол",
                           n = "Случаев",
                           percent_diag = "% в группе",
                           percent_overall = "% всего")
```

Диагноз	Пол	Случаев	% в группе	% всего
lymphoid	Женский	15	50.00000	23.4375
lymphoid	Мужской	15	50.00000	23.4375
myeloid	Женский	13	46.42857	20.3125
myeloid	Мужской	15	53.57143	23.4375
	Женский	2	33.33333	3.1250
	Мужской	4	66.66667	6.2500
Всего	Всего	64		100.0000

Рисунок 8.55. Таблица *flextable* после переименования заголовков столбцов

Для лучшего визуального восприятия можно разделить заголовки на два уровня с помощью функции *separate_header()*. Данная функция по умолчанию разделяет названия столбцов по символам точки или нижнего подчеркивания. В данном случае можно получить блок столбцов *percent* из столбцов *percent_diag* и *percent_overall*. В итоге в созданный блок *percent* будет входить два столбца: *diag* и *overall*. При необходимости можно задать другие символы, по которым будет происходить разделение столбцов с помощью параметра *split*. После применения функции *separate_header()* для задания заголовков нужно использовать функцию *labelizer()*.

```
ft <- ft %>% separate_header(split = "_") %>%
  labelizer(part = "header",
            labels = c("diagnosis" = "Диагноз",
                      "sex" = "Пол",
                      "n" = "Случаев",
                      "percent" = "Процент",
                      "diag" = "в группе",
                      "overall" = "всего"))
ft
```

		Процент		
Диагноз	Пол	Случаев	в группе	всего
lymphoid	Женский	15	50.00000	23.4375
lymphoid	Мужской	15	50.00000	23.4375
myeloid	Женский	13	46.42857	20.3125
myeloid	Мужской	15	53.57143	23.4375
	Женский	2	33.33333	3.1250
	Мужской	4	66.66667	6.2500
Всего	Всего	64		100.0000

Рисунок 8.56. Таблица *flextable* после разделения заголовков *percent_diag* и *percent_overall*

Для наглядности также следует добавить дополнительные заголовки к таблице.

```
ft <- ft %>%
  # Подпись таблицы.
  set_caption("Типы диагнозов") %>%
  # Добавление строки заголовка.
  add_header_lines("Вторая строка заголовка") %>%
  # Добавление строки примечания.
  add_footer_lines("На основании данных из набора cytomegalovirus")
ft
```

Типы диагнозов				
Вторая строка заголовка				
		Процент		
Диагноз	Пол	Случаев	в группе	всего
lymphoid	Женский	15	50.00000	23.4375
lymphoid	Мужской	15	50.00000	23.4375
myeloid	Женский	13	46.42857	20.3125
myeloid	Мужской	15	53.57143	23.4375
	Женский	2	33.33333	3.1250
	Мужской	4	66.66667	6.2500
Всего	Всего	64		100.0000

На основании данных из набора cytomegalovirus

Рисунок 8.57. Таблица *flextable* после добавления дополнительных заголовков

В рекомендациях по оформлению таблиц в разделе про единицы измерения рассматривалась ситуация, когда их указывали в сносках. В данном примере также следует добавить такие комментарии к столбцам с процентами с помощью функции `footnote()` - они попадут в раздел footer примечаний таблицы.

```
ft <- ft %>%
  # Комментарий к процентам в группе.
  footnote(i = 3, j = 4, part = "header",
            ref_symbols = "1", value = as_paragraph("Доля пациента в рамках
диагноза"))
  # Комментарий к процентам всего.
  footnote(i = 3, j = 5, part = "header",
            ref_symbols = "2", value = as_paragraph("Доля пациентов в рамках
исследования"))
ft
```

Типы диагнозов					
Вторая строка заголовка					
Процент					
Диагноз	Пол	Случаев	в группе ¹	всего ²	
lymphoid	Женский	15	50.00000	23.4375	
lymphoid	Мужской	15	50.00000	23.4375	
myeloid	Женский	13	46.42857	20.3125	
myeloid	Мужской	15	53.57143	23.4375	
	Женский	2	33.33333	3.1250	
	Мужской	4	66.66667	6.2500	
Всего	Всего	64		100.0000	

На основании данных из набора cytomegalovirus

¹Доля пациента в рамках диагноза

²Доля пациентов в рамках исследования

Рисунок 8.58. Таблица *flextable* после добавления сносок

Форматирование значений

Для улучшения восприятия данных таблицы можно модифицировать представление значений в строках таблицы. Для этого можно применять функцию `set_formatter()`, которая позволяет применить функцию к любому столбцу таблицы. Например, с

помощью функции `str_to_title()` из пакета `stringr` можно изменить написание диагнозов таким образом, чтобы их названия начинались с заглавной буквы. Есть и набор методов для определенных типов данных, которые могут применять настройки для всех столбцов таблицы определенного типа: чисел, строк, и так далее — `colformat_char()`, `colformat_date()`, `colformat_datetime()`, `colformat_image()`, `colformat_int()`, `colformat_lgl()`, `colformat_num()`, `set_formatter()`. Каждый из них имеет свои набор параметров. В представленном примере все дробные значения являются процентами, поэтому следует округлить значения до двух чисел после запятой и добавить символ процента для наглядности.

```
ft <- ft %>%
  set_formatter(diagnosis = str_to_title) %>%
  colformat_double(digits = 2, suffix = "%")
ft
```

Типы диагнозов				
Вторая строка заголовка				
Процент				
Диагноз	Пол	Случаев	в группе ¹	всего ²
Lymphoid	Женский	15	50.00%	23.44%
Lymphoid	Мужской	15	50.00%	23.44%
Myeloid	Женский	13	46.43%	20.31%
Myeloid	Мужской	15	53.57%	23.44%
	Женский	2	33.33%	3.12%
	Мужской	4	66.67%	6.25%
Всего	Всего	64		100.00%

На основании данных из набора cytomegalovirus

¹Доля пациента в рамках диагноза

²Доля пациентов в рамках исследования

Рисунок 8.59. Таблица `flextable` после округления дробных чисел и добавления символа “%”

Ширина столбцов

После того, как назначены заголовки столбцов и определены форматы вывода значений, общий вид таблицы уже практически сформирован. На этом этапе можно задать ширину столбцов. Проще всего использовать функцию `autofit()`, которая автоматически подберет нужную ширину на основе данных.

```
ft <- ft %>% autofit()
```

Типы диагнозов				
Вторая строка заголовка				
Диагноз	Пол	Случаев	Процент	
			в группе ¹	всего ²
Lymphoid	Женский	15	50.00%	23.44%
Lymphoid	Мужской	15	50.00%	23.44%
Myeloid	Женский	13	46.43%	20.31%
Myeloid	Мужской	15	53.57%	23.44%
	Женский	2	33.33%	3.12%
	Мужской	4	66.67%	6.25%
Всего	Всего	64	100.00%	

На основании данных из набора cytomegalovirus

¹Доля пациента в рамках диагноза

²Доля пациентов в рамках исследования

Рисунок 8.60. Таблица *flextable* после автоматического подбора ширины столбцов

Обычно этого достаточно, но порой необходимо дополнительно выделить отдельные столбцы, поэтому ширину столбцов можно задать индивидуально. В этом случае ширину столбцов можно указать с помощью функции *width()*. Параметр *j* принимает список столбцов, для которых нужно модифицировать ширину, *width* - значение ширины, *unit* - единицы измерения (по умолчанию используются дюймы *in*, но привычнее использовать сантиметры *cm* или миллиметры *mm*)

```
ft %>%
  width(j = c(1,2), width = 3, unit = "cm") %>%
  width(j = c(3,4,5), width = 2, unit = "cm")
```

Стоит также учитывать, что основная цель таблиц *flextable* вывод табличных данных в отчетах, поэтому можно задать способы размещения таблицы в документе с помощью функции *set_table_properties()*. В ней можно отдельно определить особенности вывода таблиц в форматах *html*, *word*, *pdf*, а также задать ширину таблицы относительно документа (значения от 0 до 1) и автоматическое определение ширины столбцов с помощью параметра *layout*.

```
ft %>% set_table_properties(width = 1, layout = "autofit")
```

Шрифты и выравнивание

Для дополнительного выделения значимых элементов можно поработать со шрифтами: выделить заголовки жирным текстом, примечания курсивом, сделать некоторые подписи крупнее, повернуть значения под углом. Для выполнения данных манипуляций используются функции: *font()*, *fontsize()*, *italic()*, *bold()*, *rotate()*.

Во всех функциях связанных с визуальным оформлением применяется сходный принцип определения элементов, для которых нужно применить изменения — необходимо использовать набор из трех параметров:

- i - список строк, можно перечислять как по номерам, так и задавать условия отбора по значениям столбцов;
- j - список столбцов, можно обращаться как по номерам, так и по имени, а также выбирать по условию;
- $part$ - часть таблицы (all - вся таблица, $body$ - тело таблицы, $header$ - заголовки, $footer$ - примечания).

Эти параметры можно комбинировать — использовать по одному или сразу все вместе, чтобы добиться желаемого результата.

Для примера форматируемая таблица будет изменена следующим образом:

- заголовок и примечания таблицы - шрифт Times New Roman;
- примечания - маленький шрифт и курсивом;
- заголовки столбцов - выделение жирным шрифтом;
- названия диагнозов - выделение курсивом;
- строка итогов - выделение жирным.

```
ft <- ft %>%
  # Выбор первой строки заголовка.
  font(fontname = "Times New Roman", i = 1, part = "header") %>%
  # Оформление примечаний.
  font(fontname = "Times New Roman", part = "footer") %>%
  fontsize(size = 9, part = "footer") %>%
  italic(part = "footer") %>%
  # Заголовки столбцов включают две строки.
  bold(i = c(2,3), part = "header") %>%
  # Выделение столбца диагнозов курсивом.
  italic(j = "diagnosis", part = "body") %>%
  # Выделение строки итогов на основе значения ячейки.
  bold(i = ~ diagnosis == "Всего", part = "body")
ft
```

Типы диагнозов				
Вторая строка заголовка				
Процент				
Диагноз	Пол	Случаев	в группе ¹	всего ²
<i>Lymphoid</i>	Женский	15	50.00%	23.44%
<i>Lymphoid</i>	Мужской	15	50.00%	23.44%
<i>Myeloid</i>	Женский	13	46.43%	20.31%
<i>Myeloid</i>	Мужской	15	53.57%	23.44%
	Женский	2	33.33%	3.12%
	Мужской	4	66.67%	6.25%
Всего	Всего	64		100.00%

*На основании данных из набора *cytomegalovirus**

¹*Доля пациента в рамках диагноза*

²*Доля пациентов в рамках исследования*

Рисунок 8.61. Таблица *flextable* после форматирования согласно условиям

Теперь следует выровнять содержимое ячеек. Есть возможность применить как "пакетное" выравнивание всех текстовых (функция *align_text_col()*) или нетекстовых (функция *align_nottext_col()*) столбцов, так и задать индивидуальные настройки для выравнивания по вертикали *valign()* и горизонтали *align()*. Элементы, для которых будет применяться выравнивание, также будут определяться комбинацией параметров *i, j, part*.

Сначала следует определить общее выравнивание для таблицы, чтобы текстовые столбцы выравнивались по левому краю, а нетекстовые по правому (на самом деле данные параметры используются по умолчанию).

```
ft <- ft %>%
  # Общее выравнивание таблицы.
  align_nottext_col(align = "right") %>%
  align_text_col(align = "left") %>%
  # Заголовки по вертикали выравниваются по центру.
  valign(valign = "center", part = "header") %>%
  # Заголовок Процент выравнивается по горизонтали по центру.
  align(i = 2, j = 4, align = "center", part = "header") %>%
  # Примечание выравнивается по правому краю.
  align(align = "right", part = "footer") %>%
  # Комментарии-сноски выравниваются по левому краю.
  align(i = c(2,3), align = "Left", part = "footer") %>%
  # Количество случаев выравнивается по горизонтали по центру.
  align(align = "center", j = "n", part = "body") %>%
  align(i = 2, j = "n", align = "center", part = "header")
ft
```

Типы диагнозов				
Вторая строка заголовка				
Диагноз	Пол	Случаев	Процент	
			в группе ¹	всего ²
<i>Lymphoid</i>	Женский	15	50.00%	23.44%
<i>Lymphoid</i>	Мужской	15	50.00%	23.44%
<i>Myeloid</i>	Женский	13	46.43%	20.31%
<i>Myeloid</i>	Мужской	15	53.57%	23.44%
	Женский	2	33.33%	3.12%
	Мужской	4	66.67%	6.25%
Всего	Всего	64		100.00%

На основании данных из набора сутомегаловирус

¹Доля пациента в рамках диагноза

²Доля пациентов в рамках исследования

Рисунок 8.62. Таблица *flextable* после выравнивания согласно условиям

Можно заметить, что в примечаниях таблицы образовались слишком большие отступы между строк. Сделать примечания компактнее можно с помощью функции *line_spacing()*.

```
ft <- ft %>% line_spacing(part = "footer", space = 0.25)
ft
```

Типы диагнозов				
Вторая строка заголовка				
Диагноз	Пол	Случаев	Процент	
			в группе ¹	всего ²
<i>Lymphoid</i>	Женский	15	50.00%	23.44%
<i>Lymphoid</i>	Мужской	15	50.00%	23.44%
<i>Myeloid</i>	Женский	13	46.43%	20.31%
<i>Myeloid</i>	Мужской	15	53.57%	23.44%
	Женский	2	33.33%	3.12%
	Мужской	4	66.67%	6.25%
Всего	Всего	64		100.00%

На основании данных из набора cütomegalovirus

¹Доля пациента в рамках диагноза

²Доля пациентов в рамках исследования

Рисунок 8.63. Таблица *flextable* с компактными примечаниями

Объединение ячеек

В рекомендациях по оформлению таблиц отмечалась важность группировки данных. Одним из способов была группировка ячеек с повторяющимися данными. В представленном примере повторяются значения в столбце *Диагноз*, а также значение *Всего* в строке итогов. Для объединения ячеек используется серия функций *merge_**:

- *merge_at()* - объединяет область, определяемую параметрами *i, j, part*;
- *merge_h()* - объединяет ячейки по горизонтали в строках;
- *merge_v()* - объединяет ячейки по вертикали в столбцах.

```
ft %>%
# Объединение ячеек с диагнозами.
merge_v(j = "diagnosis") %>%
# Объединение ячеек в строке итогов.
merge_at(i = ~ diagnosis == "Всего", j = c(1:2), part = "body")
ft
```

Типы диагнозов				
Вторая строка заголовка				
Диагноз	Пол	Случаев	Процент	
			в группе ¹	всего ²
<i>Lymphoid</i>	Женский	15	50.00%	23.44%
	Мужской	15	50.00%	23.44%
<i>Myeloid</i>	Женский	13	46.43%	20.31%
	Мужской	15	53.57%	23.44%
	Женский	2	33.33%	3.12%
	Мужской	4	66.67%	6.25%
Всего		64		100.00%

На основании данных из набора cytomegalovirus

¹*Доля пациента в рамках диагноза*
²*Доля пациентов в рамках исследования*

Рисунок 8.64. Таблица *flextable* с объединенными строками в столбце с диагнозами

По умолчанию значения в объединенных ячейках выравниваются по вертикали по центру. В данном случае следует выровнять столбец диагнозов по вертикали по верхнему краю.

```
ft <- ft %>% valign(j = 1, valign = "top", part = "body")
ft
```

Типы диагнозов				
Вторая строка заголовка				
Диагноз	Пол	Случаев	Процент	
			в группе ¹	всего ²
<i>Lymphoid</i>	Женский	15	50.00%	23.44%
	Мужской	15	50.00%	23.44%
<i>Myeloid</i>	Женский	13	46.43%	20.31%
	Мужской	15	53.57%	23.44%
	Женский	2	33.33%	3.12%
	Мужской	4	66.67%	6.25%
Всего		64	100.00%	

На основании данных из набора cytomegalovirus

¹*Доля пациента в рамках диагноза*

²*Доля пациентов в рамках исследования*

Рисунок 8.65. Таблица *flextable* с объединенными строками в столбце с диагнозами с выравниванием по верхнему краю

Выделение границ

Несмотря на то, что строки были сгруппированы по диагнозам, может потребоваться дополнительно отделить группы с помощью линий. Также для наглядности следует отделить строку итогов от основного блока таблицы. Для рисования границ таблиц существует большое количество функций следующих разновидностей:

- *border_** - для внутренних и внешних границ блоков;
- *vline_** - для вертикальных линий;
- *hline_** - для горизонтальных линий;
- *surround* - наиболее универсальная функция, которая позволяет задать как внешние, так и внутренние границы.

Все представленные функции работают с областью определенной параметрами *i*, *j*, *part*, а также с параметром *border*, который определяет какая именно граница будет нарисована: тип линии, ее толщина и т.д. Границы определяются с помощью функции *fp_border()* из пакета *officer*, и следующих параметров:

- *color* - цвет границы. Можно использовать как шестнадцатеричное представление цвета #1A1A1A, так и текстовое описание *gray10*;
- *style* - тип линии границы, допустимые названия могут отличаться в зависимости от формата вывода, но наиболее популярные и универсальные стили *none*, *solid*, *double*, *dotted*, *dashed*;
- *width* - толщина границы.

```
# Определение линии.
line_black2 = fp_border(color="black", width = 2)
```

```

Line_black1 = fp_border(color="black", width = 1)
line_gray = fp_border(color="gray", width = 1)
# Рисование линии.
ft <- ft %>%
  # Горизонтальная линия для заголовков столбцов.
  hline(part = "header", border = Line_black1) %>%
  # Горизонтальная линия для тела таблицы.
  hline(part = "body", border = line_gray) %>%
  # Вертикальная линия для столбца с процентами.
  surround(i = 2, j = 3, part = "header", border.right = Line_black1 ) %>%
  # Вертикальная линия для отделения процентов в теле таблицы.
  surround(j = 3, part = "body", border.right = Line_black1 ) %>%
  # Горизонтальная линия для строки итогов.
  surround(i = ~ diagnosis == "Всего", border.top = Line_black2, border.bottom =
Line_black2 )
ft

```

Типы диагнозов				
Диагноз	Пол	Случаев	Процент	
			в группе ¹	всего ²
<i>Lymphoid</i>	Женский	15	50.00%	23.44%
	Мужской	15	50.00%	23.44%
<i>Myeloid</i>	Женский	13	46.43%	20.31%
	Мужской	15	53.57%	23.44%
	Женский	2	33.33%	3.12%
	Мужской	4	66.67%	6.25%
Всего		64	100.00%	

На основании данных из набора cutmegalovirus

¹Доля пациента в рамках диагноза

²Доля пациентов в рамках исследования

Рисунок 8.66. Таблица *flextable* с выделением с помощью линий

Окрашивание элементов

Для выделения элементов таблицы цветом можно использовать следующие функции:

- *bg()* - задает фоновый цвет;
- *color()* - задает цвет текста;
- *highlight()* - задает фоновый цвет текста.

Все представленные функции работают с областью определенной параметрами *i*, *j*, *part*, поэтому можно окрашивать не только целиком строки и столбцы, но и отдельные значения, отбираемые по условию. Цвета окрашивания можно задавать как в шестнадцатеричном формате, так и с помощью зарезервированных слов.

```

ft <- ft %>%
  # Фон заголовков столбцов сделать серым.
  bg(bg = "grey95", part = "header") %>%
  # Подсветить строки, где доля диагноза более 50%.
  bg(i = ~ percent_diag > 50, bg = "#ffa50040") %>%
  # Выделить значения, где доля диагноза более 50%.
  color(i = ~ percent_diag > 50, j = 4, color = "red") %>%
  # Выделить количество, где доля диагноза более 50%.
  highlight(i = ~ percent_diag > 50, j = 3, color = "pink")

```

ft

Типы диагнозов				
Вторая строка заголовка				
Диагноз	Пол	Случаев	Процент	
			в группе ¹	всего ²
<i>Lymphoid</i>	Женский	15	50.00%	23.44%
	Мужской	15	50.00%	23.44%
<i>Myeloid</i>	Женский	13	46.43%	20.31%
	Мужской	15	53.57%	23.44%
	Женский	2	33.33%	3.12%
	Мужской	4	66.67%	6.25%
Всего		64	100.00%	

На основании данных из набора *cytomegalovirus*

¹Доля пациента в рамках диагноза

²Доля пациентов в рамках исследования

Рисунок 8.67. Таблица *flextable* с выделением значений по условию

Создание собственной темы

В результате всех манипуляций полный текст для оформления таблицы выглядит следующим образом.

```

ft_full <- data %>% flextable() %>%
  # Разделение столбцов.
  separate_header(split = "_") %>%
  labelizer(part = "header",
            labels = c("diagnosis" = "Диагноз",
                      "sex" = "Пол",
                      "n" = "Случаев",
                      "percent" = "Процент",
                      "diag" = "в группе",
                      "overall" = "всего")) %>%
  # Указание заголовков и примечаний для таблицы.
  set_caption("Типы диагнозов") %>%
  add_header_lines("Вторая строка заголовка") %>%
  add_footer_lines("На основании данных из набора cytomegalovirus") %>%

```

```

footnote(i = 3, j = 4, part = "header",
         ref_symbols = "1", value = as_paragraph("Доля пациента в рамках
диагноза")) %>%
footnote(i = 3, j = 5, part = "header",
         ref_symbols = "2", value = as_paragraph("Доля пациентов в рамках
исследования")) %>%
# Форматирование значений.
set_formatter(diagnosis = str_to_title) %>%
colformat_double(digits = 2, suffix = "%") %>%
# Шрифты.
font(fontname = "Times New Roman", i = 1, part = "header") %>%
font(fontname = "Times New Roman", part = "footer") %>%
fontsize(size = 9, part = "footer") %>%
italic(part = "footer") %>%
bold(i = c(2,3), part = "header") %>%
italic(j = "diagnosis", part = "body") %>%
bold(i = ~ diagnosis == "Всего", part = "body") %>%
# Выравнивание.
align_nottext_col(align = "right") %>%
align_text_col(align = "left") %>%
valign(valign = "center", part = "header") %>%
align(i = 2, j = 4, align = "center", part = "header") %>%
align(align = "right", part = "footer") %>%
align(i = c(2,3), align = "left", part = "footer") %>%
align(align = "center", j = "n", part = "body") %>%
align(i = 2, j = "n", align = "center", part = "header") %>%
line_spacing(part = "footer", space = 0.25) %>%
# Объединение ячеек.
merge_v(j = "diagnosis") %>%
merge_at(i = ~ diagnosis == "Всего", j = c(1:2), part = "body") %>%
valign(j = 1, valign = "top", part = "body") %>%
# Выделение границ.
hline(part = "header", border = fp_border(color="black", width = 1)) %>%
hline(part = "body", border = fp_border(color="gray", width = 1)) %>%
surround(i = 2, j = 3, part = "header",
          border.right = fp_border(color="black", width = 1)) %>%
surround(j = 3, part = "body",
          border.right = fp_border(color="black", width = 1)) %>%
surround(i = ~ diagnosis == "Всего",
          border.top = fp_border(color="black", width = 2),
          border.bottom = fp_border(color="black", width = 2)) %>%
# Окрашивание элементов.
bg(bg = "grey95", part = "header") %>%
bg(i = ~ percent_diag > 50, bg = "#ffa50040") %>%
color(i = ~ percent_diag > 50, j = 4, color = "red") %>%
highlight(i = ~ percent_diag > 50, j = 3, color = "pink") %>%
# Автоматическая ширина столбцов.
autofit()
ft_full

```

Отчет обычно включает множество таблиц, а так как они должны быть выполнены в едином стиле, то придется для каждой таблицы писать объемный код для оформления, что кажется неоптимальным и трудозатратным. Проблема усугубляется, когда требуется изменить оформление таблиц — в этом случае придется исправлять код оформления для каждой таблицы.

Чтобы избежать таких сложностей существует два подхода. Первый — использовать функцию `set_flextable_defaults()` которая позволяет задать настройки по умолчанию для

всех таблиц, создаваемых пакетом *flextable* в рамках текущей сессии. С помощью данной функции можно определить правила оформления значений, цвета заливки, размеры и стили шрифтов, величину отступов и т.д. После ее применения все таблицы будут оформляться согласно установленным параметрам. После использования данной функции каждую таблицу можно модифицировать дополнительно. Чтобы сбросить установленные параметры на значения по умолчанию можно использовать функцию *init_flextable_defaults()*.

Однако не все настройки оформления можно установить с помощью значений по умолчанию, поэтому дополнительно можно применить второй способ — создать собственную функцию, которая будет применять нужные стили оформления. Созданную функцию можно применять к каждой таблице в рамках конвейера обработки данных. Для реализации этого подхода из скрипта работы с таблицей необходимо выделить функции, которые могут работать независимо от характера представленных данных. Например, в итоговом отчете фигурируют две таблицы: *data* и *data_age*.

```
data
# A tibble: 7 × 5
  diagnosis sex      n percent_diag percent_overall
  <chr>     <chr>   <int>        <dbl>        <dbl>
1 Lymphoid Женский  15       50          23.4
2 Lymphoid Мужской  15       50          23.4
3 myeloid Женский  13       46.4        20.3
4 myeloid Мужской  15       53.6        23.4
5 NA        Женский  2        33.3        3.12
6 NA        Мужской  4        66.7        6.25
7 Всего     Всего    64       NA          100

data_age
# A tibble: 7 × 6
  diagnosis sex      n min_age avg_age max_age
  <chr>     <chr>   <int>    <dbl>    <dbl>    <dbl>
1 Lymphoid Женский  15       39       55.6      64
2 Lymphoid Мужской  15       35       56.2      67
3 myeloid Женский  13       33       49.3      63
4 myeloid Мужской  15       34       52.6      62
5 NA        Женский  2        29       37.5      46
6 NA        Мужской  4        36       43.5      48
7 Всего     Всего    64       29       52.4      67
```

Следует отметить, что представленные таблицы имеют схожее содержимое: в каждой таблице есть столбцы с диагнозом, полом, количеством и строка итогов. Учитывая одинаковое наименование столбцов (*diagnosis*, *sex*, *n*) можно создать функцию для оформления подобных таблиц. Далее представлена функция для оформления представленных выше таблиц: в ней описываются общие команды для форматирования, а также заданы заголовок и примечания для таблицы.

```
format_my_table <- function(data, caption, comment) {
  data %>% flextable() %>% theme_booktabs() %>%
    # Разделение столбцов.
    separate_header(split = "_") %>%
    labelizer(part = "header",
              Labels = c("diagnosis" = "Диагноз",
                        "sex" = "Пол",
```

```

    "n" = "Случаев"
)) %>%
# Добавление заголовка и примечания.
set_caption(caption) %>%
add_footer_lines(comment) %>%
# Форматирование значений.
set_formatter(diagnosis = str_to_title) %>%
# Шрифты.
font(fontname = "Times New Roman", part = "footer") %>%
fontsize(size = 9, part = "footer") %>%
italic(part = "footer") %>%
bold(part = "header") %>%
italic(j = "diagnosis", part = "body") %>%
bold(i = ~ diagnosis == "Всего", part = "body") %>%
# Выравнивание.
align_nottext_col(align = "right") %>%
align_text_col(align = "Left") %>%
# Выравнивание заголовков.
valign(valign = "center", part = "header") %>%
align(i = 1, j = 4, align = "center", part = "header") %>%
# Выравнивание количества пациентов.
align(j = "n", align = "center", part = "header") %>%
align(align = "center", j = "n", part = "body") %>%
# Выравнивание примечаний.
align(align = "right", part = "footer") %>%
line_spacing(part = "footer", space = 0.25) %>%
# Объединение ячеек.
merge_v(j = "diagnosis") %>%
merge_at(i = ~ diagnosis == "Всего", j = c(1:2), part = "body") %>%
valign(j = 1, valign = "top", part = "body") %>%
# Выделение границ.
hline(part = "header", border = fp_border(color="black", width = 1)) %>%
hline(part = "body", border = fp_border(color="gray", width = 1)) %>%
surround(j = 3, part = "header",
          border.right = fp_border(color="black", width = 1)) %>%
surround(j = 3, part = "body",
          border.right = fp_border(color="black", width = 1)) %>%
surround(i = ~ diagnosis == "Всего",
          border.top = fp_border(color="black", width = 2),
          border.bottom = fp_border(color="black", width = 2)) %>%
# Окрашивание элементов.
bg(bg = "grey95", part = "header")
}

```

Теперь необходимо применить созданную функцию *format_my_table()* к данным таблицы.

```

ft_data <- data %>% format_my_table(
  caption = "Типы диагнозов",
  comment = "На основании данных из набора cytomegalovirus")
ft_data

```

Типы диагнозов				
Диагноз	Пол	Случаев	percent	
			diag	overall
<i>Lymphoid</i>	Женский	15	50.00000	23.4375
	Мужской	15	50.00000	23.4375
<i>Myeloid</i>	Женский	13	46.42857	20.3125
	Мужской	15	53.57143	23.4375
	Женский	2	33.33333	3.1250
	Мужской	4	66.66667	6.2500
Всего		64	100.0000	

На основании данных из набора cytomegalovirus

Рисунок 8.68. Таблица *flextable* с применением созданной пользователем функции *format_my_table()*

```
ft_data_age <- data_age %>% format_my_table(
  caption = "Возраст пациентов",
  comment = "На основании данных из набора cytomegalovirus"
)
ft_data_age
```

Возраст пациентов					
Диагноз	Пол	Случаев	age		
			min	avg	max
<i>Lymphoid</i>	Женский	15	39	55.60000	64
	Мужской	15	35	56.20000	67
<i>Myeloid</i>	Женский	13	33	49.30769	63
	Мужской	15	34	52.60000	62
	Женский	2	29	37.50000	46
	Мужской	4	36	43.50000	48
Всего		64	29	52.43750	67

На основании данных из набора cytomegalovirus

Рисунок 8.69. Таблица *flextable* с возрастом пациентов с применением созданной пользователем функции *format_my_table()*

В результате обе таблицы оформлены в одном стиле и нет необходимости в дублировании функций оформления для каждой из таблиц. После применения функции *format_my_table()* можно дополнить оформление недостающими функциями.

```

ft_data %>%
  # Определение недостающих заголовков.
  labelizor(part = "header",
            labels = c("percent" = "Процент",
                      "diag" = "в группе",
                      "overall" = "всего")) %>%
  # Добавление символа процентов.
  colformat_double(digits = 2, suffix = "%") %>%
  # Указание примечаний для таблицы.
  footnote(i = 2, j = 4, part = "header",
            ref_symbols = "1", value = as_paragraph("Доля пациента в рамках
диагноза")) %>%
  footnote(i = 2, j = 5, part = "header",
            ref_symbols = "2", value = as_paragraph("Доля пациентов в рамках
исследования")) %>%
  # Выравнивание примечаний по левому краю.
  align(i = c(2,3), align = "Left", part = "footer") %>%
  # Окрашивание элементов.
  bg(bg = "grey95", part = "header") %>%
  bg(i = ~ percent_diag > 50, bg = "#ffa50040") %>%
  color(i = ~ percent_diag > 50, j = 4, color = "red") %>%
  highlight(i = ~ percent_diag > 50, j = 3, color = "pink") %>%
  # Автоматический подбор ширины столбцов.
  autofit()

```

Типы диагнозов			Процент	
Диагноз	Пол	Случаев	в группе ¹	всего ²
<i>Lymphoid</i>	Женский	15	50.00%	23.44%
	Мужской	15	50.00%	23.44%
<i>Myeloid</i>	Женский	13	46.43%	20.31%
	Мужской	15	53.57%	23.44%
	Женский	2	33.33%	3.12%
	Мужской	4	66.67%	6.25%
Всего			100.00%	

На основании данных из набора cytomegalovirus

¹Доля пациента в рамках диагноза

²Доля пациентов в рамках исследования

Рисунок 8.70. Таблица *flextable* с применением созданной пользователем функции *format_my_table()* и последующим использованием недостающих функций

В результате мы получилась таблица, идентичная той, которая была сделана ранее. Аналогично можно выполнить оформление для таблицы *ft_data_age*.

```

ft_data_age %>%
  # Оформление числовых значений.
  colformat_double(j = "age_avg", digits = 2) %>%

```

```

# Определение недостающих заголовков.
labelizer(part = "header",
          labels = c("age" = "Возраст",
                     "min" = "Мин",
                     "avg" = "Сред",
                     "max" = "Макс")) %>%
# Подбор ширины столбцов.
autofit()

```

Возраст пациентов					
Диагноз	Пол	Случаев	Возраст		
			Мин	Сред	Макс
<i>Lymphoid</i>	Женский	15	39	55.60	64
	Мужской	15	35	56.20	67
<i>Myeloid</i>	Женский	13	33	49.31	63
	Мужской	15	34	52.60	62
	Женский	2	29	37.50	46
	Мужской	4	36	43.50	48
Всего		64	29	52.44	67

На основании данных из набора *cytomegalovirus*

Рисунок 8.71. Таблица *flextable* с возрастом пациентов с применением созданной пользователем функции *format_my_table()* и последующим использованием недостающих функций

Сохранение таблицы

Обычно таблицы *flextable* создаются для встраивания в отчеты, но есть возможность и сохранить из отдельно в файл с помощью семейства функций *save_as_**.

При сохранении в форматы MS Word и PowerPoint таблица сохраняется именно как объект таблицы и с ним можно продолжать работу уже непосредственно в документе: изменять ширину столбцов, редактировать текст, перекрашивать ячейки.

```
ft %>% save_as_docx(path = "flextable.docx")
```

flextable.docx - Word

File Home Insert Design Layout References Mailings Review View Zotero Help Design Layout Tell me Share

Table Style Options

Table Styles

Borders

Painter

Border Styles

Pen Color

5 ··· 4 ··· 3 ··· 2 ··· 1 ··· 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

Типы диагнозов

Вторая строка заголовка

Диагноз	Пол	Случаев	Процент	
			в группе ¹	всего ²
<i>Lymphoid</i>	Женский	15	50.00%	23.44%
	Мужской	15	50.00%	23.44%
<i>Myeloid</i>	Женский	13	46.43%	20.31%
	Мужской	15	53.57%	23.44%
	Женский	2	33.33%	3.12%
	Мужской	4	66.67%	6.25%
Всего		64		100.00%

На основании данных из набора сутомегаловирус

¹Доля пациентов в рамках диагноза
²Доля пациентов в рамках исследования

Page 1 of 1 57 words English (United States) Accessibility: Investigate 100 %

Рисунок 8.72. Таблица *flextable*, сохраненная в MS Word

```
ft %>% save_as_pptx(path = "flextable.pptx")
```

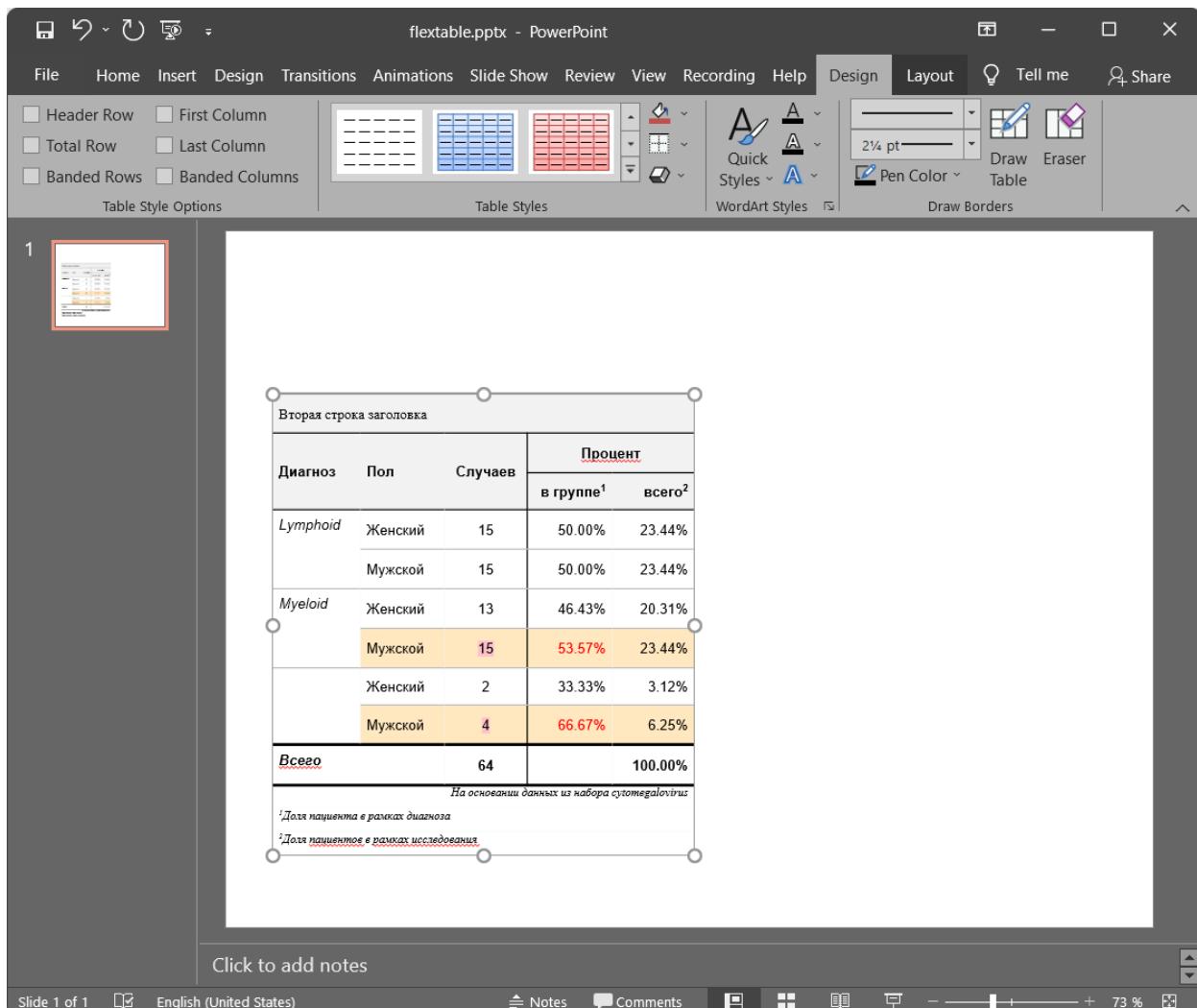


Рисунок 8.73. Таблица *flextable*, сохраненная в PowerPoint

Можно сохранить таблицу в виде изображения: векторного (SVG) или растрового (PNG). Подробнее отличия векторных и растровых изображений будут рассмотрены в следующих разделах, а пока следует отметить, что в этом случае таблица перестает быть редактируемой. Такие изображения можно использовать для вставки в различные публикации: статьи, постеры и так далее.

При сохранении таблицы как изображения PNG важно указать величину пространства в пикселях вокруг таблицы с помощью параметра *expand*, а также разрешение изображения *res*. Для разрешения работает принцип "чем больше - тем лучше", но в целом для просмотра графика на экране компьютера минимально допустимое разрешение - 96, для печати на принтере формата А4 - 300, для печати на больших форматах - 600 и выше. Для векторных изображений SVG эти параметры менее актуальны.

```
ft %>% save_as_image(path = "flextable.png", expand = 50, res = 300)
ft %>% save_as_image(path = "flextable.svg")
```

Следует отметить, что по умолчанию, при сохранении изображения фон таблицы остается прозрачным, если заранее не определен цвет заливки ячеек. Это может влиять на просмотр изображений на устройствах с темной темой.

Сохранение в формате HTML является оптимальным для встраивания таблицы в интерактивные отчеты и просмотра в браузере — это позволяет сохранить форматирование максимально таким, каким оно представлено при работе в RStudio. Кроме того, сохранение таблицы в данном формате оставляет возможность копирования текста.

```
ft %>% save_as_html(path = "flextable.html")
```

Диагноз	Пол	Случаев	Процент	
			в группе ¹	всего ²
<i>Lymphoid</i>	Женский	15	50.00%	23.44%
	Мужской	15	50.00%	23.44%
<i>Myeloid</i>	Женский	13	46.43%	20.31%
	Мужской	15	53.57%	23.44%
	Женский	2	33.33%	3.12%
	Мужской	4	66.67%	6.25%
Всего		64	100.00%	

На основании данных из набора сутомегаловирус

¹*Доля пациента в рамках диагноза*

²*Доля пациентов в рамках исследования*

Рисунок 8.74. Таблица *flextable*, сохраненная в HTML и открытая в браузере

Отображение таблицы может незначительно отличаться в зависимости от формата сохранения документа. Это связано с особенностями конечных форматов: например, не все стили границ HTML поддерживаются в Word, некоторые шрифты могут отображаться некорректно. Рекомендуется заранее определить формат отчета и настроить стили оформления таблицы в соответствии с его особенностями. Это гарантирует, что таблица будет отображаться корректно в конечном файле.

8.2.3. Оформление таблиц с пакетом *gt*

Вторым популярным пакетом для оформления таблиц является *gt*. Он в первую очередь предназначен для публикации таблиц в интерактивных отчетах в формате HTML, но также поддерживает и вывод в форматы Word и PNG. Структура создаваемого табличного объекта представлена на **рисунке 8.75**.

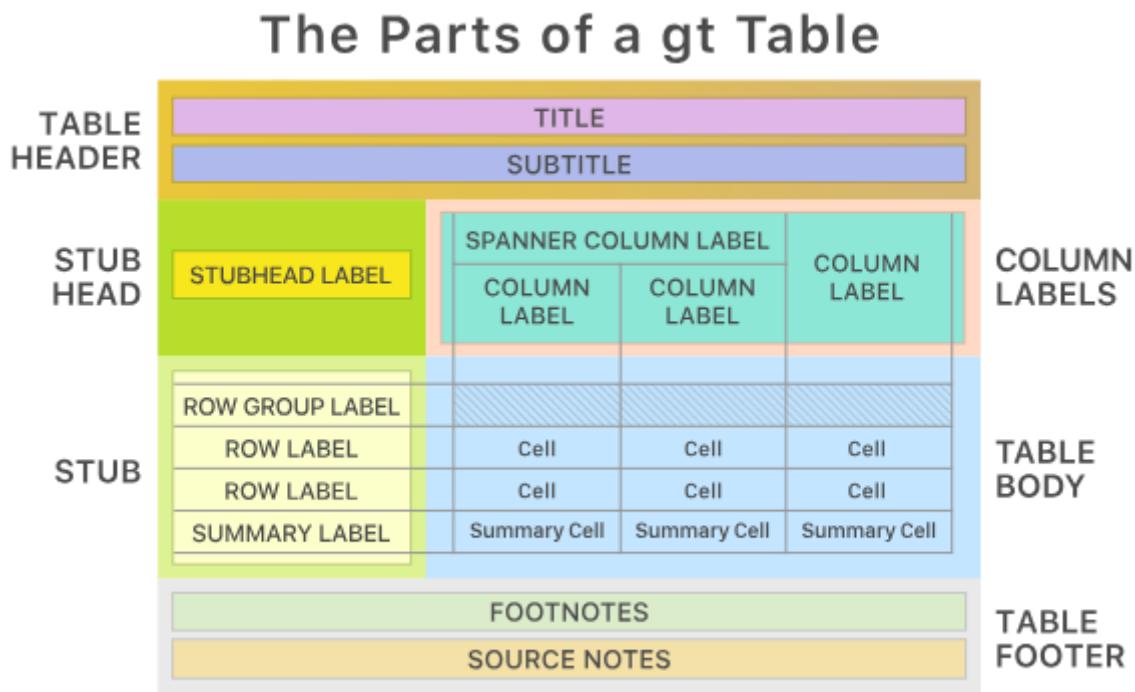


Рисунок 8.75. Общая структура таблицы *gt*

Можно заметить, что структура таблиц *gt* более сложная, по сравнению с тем, что было представлено раньше. Чтобы наглядно продемонстрировать возможности пакета *gt*, будет использована та же таблица, что и при работе с пакетом *flextable*, однако форматирование будет осуществляться с помощью функций из пакета *gt*. Создание объекта таблицы происходит с помощью функции *gt()*.

```
gt <- data %>% gt()  
gt
```

diagnosis	sex	n	percent_diag	percent_overall
lymphoid	Женский	15	50.00000	23.4375
lymphoid	Мужской	15	50.00000	23.4375
myeloid	Женский	13	46.42857	20.3125
myeloid	Мужской	15	53.57143	23.4375
NA	Женский	2	33.33333	3.1250
NA	Мужской	4	66.66667	6.2500
Всего	Всего	64	NA	100.0000

Рисунок 8.76. Таблица *gt*

После того как появился объект таблицы *gt* можно применять к нему функции, аналогично тому, как это делалось с *flextable*. Например, чтобы убрать пропущенные значения NA, можно использовать функцию *sub_missing()*, которая может заменить пропущенные значения на некоторое произвольное значение. Практически все функции, работающие с объектом *gt* принимают в качестве параметров *columns* и *rows*, которые задают область действия выбора. Если их не определять, по умолчанию действие функции распространится на всю таблицу.

```
gt %>% sub_missing(missing_text = "")
```

diagnosis	sex	n	percent_diag	percent_overall
lymphoid	Женский	15	50.00000	23.4375
lymphoid	Мужской	15	50.00000	23.4375
myeloid	Женский	13	46.42857	20.3125
myeloid	Мужской	15	53.57143	23.4375
	Женский	2	33.33333	3.1250
	Мужской	4	66.66667	6.2500
Всего	Всего	64		100.0000

Рисунок 8.77. Таблица *gt* после замены NA на пустые значения

Можно передавать как названия столбцов/номера строк, так и список с перечислением столбцов/строк. Например, чтобы заменить значение NA только в столбце *diagnosis* необходимо указать название столбца. Таким образом все пустые значения указанного столбца будут автоматически заменены.

```
gt %>% sub_missing(columns = "diagnosis", missing_text = "")
```

diagnosis	sex	n	percent_diag	percent_overall
lymphoid	Женский	15	50.00000	23.4375
lymphoid	Мужской	15	50.00000	23.4375
myeloid	Женский	13	46.42857	20.3125
myeloid	Мужской	15	53.57143	23.4375
	Женский	2	33.33333	3.1250
	Мужской	4	66.66667	6.2500
Всего	Всего	64	NA	100.0000

Рисунок 8.78. Таблица *gt* после замены NA на пустые значения только в столбце *diagnosis*

Допускается также использование выражений, которые отбирают столбцы и строки по какому-либо условию. Например, можно заменить значение NA во всех столбцах, которые начинаются на *percent_*.

```
gt %>% sub_missing(columns = starts_with("percent_"), missing_text = "")
```

diagnosis	sex	n	percent_diag	percent_overall
lymphoid	Женский	15	50.00000	23.4375
lymphoid	Мужской	15	50.00000	23.4375
myeloid	Женский	13	46.42857	20.3125
myeloid	Мужской	15	53.57143	23.4375
NA	Женский	2	33.33333	3.1250
NA	Мужской	4	66.66667	6.2500
Всего	Всего	64		100.0000

Рисунок 8.79. Таблица *gt* после замены NA на пустые значения только в столбцах с форматом названия *percent_*

Таким образом, последовательно применяя подобные функции, можно определять внешний вид таблицы. Далее будут разобраны основные приемы оформления таблиц *gt*.

Заголовки и примечания

Заголовки столбцов, переданного для обработки *data.frame*, сохраняются всегда и не разделяются как в *flextable*. Они создают объекты *column label* в созданной таблице *gt*. Процедуры "разделения" столбцов в *gt* нет, но можно задать объединяющую метку *spanner column label*, которая будет располагаться над названиями столбцов. При этом эта метка будет представлять собой отдельный независимый объект. Таким образом, с *column label* и *spanner column label* можно работать независимо. Для условного "разделения" столбцов с использованием разделителя в пакете *gt* используется функция *tab_spacer_delim*, но в данном случае будет использоваться функция *tab_spacer()*, так как она более универсальна и позволяет выбирать столбцы по условиям и сразу задать метку.

```
gt <- gt %>%
  # Определение названий столбцов.
  cols_label(diagnosis = "Диагноз", sex = "Пол", n = "Случаев",
             percent_diag = "в группе", percent_overall = "всего") %>%
  # Разделение столбцов по разделителям.
  tab_spacer(Label = "Процент", columns = starts_with("percent_"))
gt
```

Процент				
Диагноз	Пол	Случаев	в группе	всего
lymphoid	Женский	15	50.00000	23.4375
lymphoid	Мужской	15	50.00000	23.4375
myeloid	Женский	13	46.42857	20.3125
myeloid	Мужской	15	53.57143	23.4375
	Женский	2	33.33333	3.1250
	Мужской	4	66.66667	6.2500
Всего	Всего	64		100.0000

Рисунок 8.80. Таблица *gt* с группировкой столбцов с названием формата *percent_*

Для каждой таблицы можно определить заголовок *title* и подзаголовок *subtitle* — все они относятся к блоку *header*. Элементы таблицы (заголовок, подзаголовок) добавляются вверху таблицы и относятся к ней в целом. Порядок их добавления не имеет значения, так как они всегда располагаются друг под другом. Важно отметить, что таблица может иметь только один заголовок и один подзаголовок. При добавлении нескольких заголовков или подзаголовков, применяется последний добавленный элемент.

```
gt <- gt %>%
  tab_header(title = "Типы диагнозов", subtitle = "Вторая строка заголовка")
gt
```

Диагноз	Пол	Случаев в группе	всего
lymphoid	Женский	15 50.00000	23.4375
lymphoid	Мужской	15 50.00000	23.4375
myeloid	Женский	13 46.42857	20.3125
myeloid	Мужской	15 53.57143	23.4375
	Женский	2 33.33333	3.1250
	Мужской	4 66.66667	6.2500
Всего	Всего	64	100.0000

Рисунок 8.81. Таблица *gt* с заголовком и подзаголовок

При этом допускается передача не только текста, но и текста в форматах HTML и Markdown (см. [Раздел 8.2.7.](#)). Это позволяет определять стили шрифтов непосредственно в рамках форматирования HTML или Markdown, без работы со стилями таблицы.

```

gt %>%
tab_header(
  title = md("## Типы диагнозов"),
  subtitle = html("<p style='color:red;'>Вторая строка заголовка</p>")
)

```

Типы диагнозов

Вторая строка заголовка

Диагноз	Пол	Случаев	Процент	
			в группе	всего
lymphoid	Женский	15	50.00000	23.4375
lymphoid	Мужской	15	50.00000	23.4375
myeloid	Женский	13	46.42857	20.3125
myeloid	Мужской	15	53.57143	23.4375
	Женский	2	33.33333	3.1250
	Мужской	4	66.66667	6.2500
Всего	Всего	64		100.0000

Рисунок 8.82. Таблица *gt* с заголовком и подзаголовком, отформатированными непосредственно в HTML или Markdown

Примечания в таблицах *gt* могут быть двух типов: сноски *footnotes* и источники *sourcenotes*. При этом порядок добавления не важен — сноски всегда будут располагаться над источниками, но при этом сам по себе каждый из блоков может включать в себя несколько строк.

Обычно сноски относятся к каким-либо данным в строках или столбцах, поэтому при создании нужно указать параметр *location*, в котором следует определить, куда будет добавлена цифра для сноски. Сама цифра будет определяться порядком добавления сноски в таблицу.

В сносках и источниках, также как и в заголовках можно использовать HTML и Markdown.

```
gt <- gt %>%
  tab_source_note(source_note = md("На основании данных из набора
`cytomegalovirus`")) %>%
  tab_footnote(footnote = "Доля пациента в рамках диагноза",
               locations = cells_column_labels(columns = percent_diag)) %>%
  tab_footnote(footnote = "Доля пациентов в рамках исследования",
               locations = cells_column_labels(columns = percent_overall))
gt
```

Типы диагнозов

Вторая строка заголовка

Диагноз	Пол	Случаев	Процент	
			в группе ¹	всего ²
lymphoid	Женский	15	50.00000	23.4375
lymphoid	Мужской	15	50.00000	23.4375
myeloid	Женский	13	46.42857	20.3125
myeloid	Мужской	15	53.57143	23.4375
	Женский	2	33.33333	3.1250
	Мужской	4	66.66667	6.2500
Всего	Всего	64		100.0000

¹ Доля пациента в рамках диагноза

² Доля пациентов в рамках исследования

На основании данных из набора `cytomegalovirus`

Рисунок 8.83. Таблица `gt` с добавлением сносок

Форматирование значений

При форматировании значений меняется только их визуальное представление, но не сами данные. Исходя из характера данных можно использовать как специализированные функции, например `fmt_email()` для оформления адресов электронной почты, или `fmt_percent()` для процентов, так и более универсальные, например `fmt()`, где можно указать функцию обработки входящих значений.

```
gt <- gt %>%
  fmt(columns = c(1:2), fns = str_to_title) %>%
  fmt_percent(columns = starts_with("percent"), decimals = 2, scale_values =
  FALSE)
gt
```

Типы диагнозов

Вторая строка заголовка

Диагноз	Пол	Случаев	Процент	
			в группе ¹	всего ²
Lymphoid	Женский	15	50.00%	23.44%
Lymphoid	Мужской	15	50.00%	23.44%
Myeloid	Женский	13	46.43%	20.31%
Myeloid	Мужской	15	53.57%	23.44%
	Женский	2	33.33%	3.12%
	Мужской	4	66.67%	6.25%
Всего	Всего	64		100.00%

¹ Доля пациента в рамках диагноза

² Доля пациентов в рамках исследования

На основании данных из набора `cytomegalovirus`

Рисунок 8.84. Таблица `gt` с форматированием значений

Ширина столбцов

По умолчанию ширина столбцов устанавливается автоматически, но можно и задать вручную с помощью функции `col_width()`. Допускается задание ширины как в пикселях (`px(150)`), так и в процентах (`pct(45)`). Столбцы можно определять как перечислением, так и по условию отбора.

```
gt <- gt %>%
  cols_width(diagnosis:sex ~ px(100),
              n ~ px(75),
              starts_with("percent_") ~ px(90))
```

gt

Типы диагнозов				
Вторая строка заголовка				
Диагноз	Пол	Случаев	Процент	
			в группе ¹	всего ²
Lymphoid	Женский	15	50.00%	23.44%
Lymphoid	Мужской	15	50.00%	23.44%
Myeloid	Женский	13	46.43%	20.31%
Myeloid	Мужской	15	53.57%	23.44%
	Женский	2	33.33%	3.12%
	Мужской	4	66.67%	6.25%
Всего	Всего	64		100.00%

¹ Доля пациента в рамках диагноза

² Доля пациентов в рамках исследования

На основании данных из набора `cytomegalovirus`

Рисунок 8.85. Таблица `gt` с настройкой ширины столбца

Шрифты и выравнивание

Для определения шрифтов таблицы можно воспользоваться несколькими способами. Первый — задать глобальные настройки таблицы с помощью функции `tab_options()`. Она содержит в себе массу различных параметров, которые определяют стиль отображения таблицы. Названия параметров обычно начинаются с блока, за который они отвечают, а потом характеристика. Примеры названий блоков: `table`, `heading`, `column_labels`, `table_body` и т.д.

Например, чтобы изменить шрифт глобально для таблицы можно определить следующие параметры:

- `table.font.names`
- `table.font.size`
- `table.font.weight`
- `table.font.style`
- `table.font.color`

А затем уточнить непосредственное начертание заголовков столбцов с помощью параметров:

- `column_labels.font.size`
- `column_labels.font.weight`
- `column_labels.text_transform`

Таким образом, можно определить каждый из блоков таблицы.

```
gt <- gt %>%
  tab_options(
    # Таблица в целом.
    table.font.names = 'Times New Roman',
    table.font.color = 'black',
    table.font.size = 14,
    # Заголовок и подзаголовок.
    heading.title.font.size = 18,
    heading.title.font.weight = "bold",
    heading.subtitle.font.size = 16,
    # Заголовки столбцов.
    column_labels.font.weight = 'bold',
    # Сноски.
    footnotes.font.size = 12,
    # Примечания.
    source_notes.font.size = 12
  )
gt
```

Типы диагнозов

Вторая строка заголовка

Диагноз	Пол	Случаев	Процент	
			в группе ¹	всего ²
Lymphoid	Женский	15	50.00%	23.44%
Lymphoid	Мужской	15	50.00%	23.44%
Myeloid	Женский	13	46.43%	20.31%
Myeloid	Мужской	15	53.57%	23.44%
	Женский	2	33.33%	3.12%
	Мужской	4	66.67%	6.25%
Всего	Всего	64		100.00%

¹ Доля пациента в рамках диагноза

² Доля пациентов в рамках исследования

На основании данных из набора *cytomegalovirus*

Рисунок 8.86. Таблица *gt* с настройкой шрифтов

Для более индивидуальной настройки уже придется использовать функцию *tab_style()*. Она позволяет задать индивидуальные настройки отображения отдельных элементов таблицы вплоть до ячеек. Функция принимает два аргумента:

- *location* - в каком месте меняем оформление;
- *style* - как именно меняем оформление.

Для определения элементов, которые подвергнутся изменениям можно использовать следующие вспомогательные функции, например:

- *cells_body()*
- *cells_column_labels()*
- *cells_column_spacers()*
- *cells_footnotes()*

- *cells_title()*

Стили оформления также определяются вспомогательными функциями:

- *cell_borders()* - для рисования границ ячеек;
- *cell_fill()* - для заливки цветов;
- *cell_text()* - для работы со шрифтами.

С помощью последовательного применения функции *tab_style()* с различными комбинациями параметров *location* и *style* можно переопределять стилевое оформление, установленное с помощью *tab_options()*.

```
gt <- gt %>%
  # Ячейки таблицы.
  tab_style(
    style = cell_text(font = "Arial", size = 12),
    locations = list(
      cells_body(),
      cells_column_labels(),
      cells_column_spanners()
    ) %>%
    # Столбец Диагноз.
    tab_style(
      style = cell_text(font = "Times New Roman", style = 'italic', size = 16,
v_align = "middle"),
      locations = cells_body(columns = diagnosis)
    ) %>%
    # Стока итогов.
    tab_style(
      style = cell_text(font = "Arial", weight = 'bold', style = 'normal', size =
12),
      locations = cells_body(rows = Length(diagnosis))
    ) %>%
    # Сноски и источники.
    tab_style(
      style = cell_text(style = 'italic'),
      locations = list(
        cells_footnotes(),
        cells_source_notes()
      ) %>%
    # Выравнивание по центру для столбца Случаев.
    tab_style(
      style = cell_text(align = "center"),
      locations = list(
        cells_body(columns = n),
        cells_column_labels(columns = n)
      ) %>%
    # Выравнивание по левому краю для подзаголовка.
    tab_style(
      style = cell_text(align = "left"),
      locations = cells_title(c("subtitle")))
    ) %>%
    # Выравнивание по правому краю для источника.
    tab_style(
      style = cell_text(align = "right"),
      locations = cells_source_notes()
    ) %>%
```

```

# Выравнивание по центру для заголовков столбцов.
tab_style(
  style = cell_text(v_align = "middle"),
  locations = list(
    cells_column_labels(),
    cells_column_spacers()
  )
)

```

gt

Типы диагнозов

Вторая строка заголовка

Диагноз	Пол	Случаев	Процент	
			в группе ¹	всего ²
<i>Lymphoid</i>	Женский	15	50.00%	23.44%
<i>Lymphoid</i>	Мужской	15	50.00%	23.44%
<i>Myeloid</i>	Женский	13	46.43%	20.31%
<i>Myeloid</i>	Мужской	15	53.57%	23.44%
	Женский	2	33.33%	3.12%
	Мужской	4	66.67%	6.25%
Всего	Всего	64		100.00%

¹ Доля пациента в рамках диагноза

² Доля пациентов в рамках исследования

На основании данных из набора *cytomegalovirus*

Рисунок 8.87. Таблица *gt* с точной настройкой шрифтов и выравниванием для каждой части таблицы

Группировка строк

Для группировки блоков строк в пакете *gt* используется специальный подход. В самом начале при создании таблицы можно определить группирующий столбец и названия строк. Это позволяет уже в рамках непосредственно объекта *gt*-таблицы работать с группами, считать по ним агрегирующие функции и так далее. Названия групп, имена

строк внутри групп, агрегации описываются в блоках *row group label*, *row label*, *summary label* соответственно. Введение группировки сильно влияет на внешний вид таблицы. При добавлении столбца группировки появляется строка с названием группы.

```
data %>% gt(groupname_col = "diagnosis")
```

sex	n	percent_diag	percent_overall
lymphoid			
Женский	15	50.00000	23.4375
Мужской	15	50.00000	23.4375
myeloid			
Женский	13	46.42857	20.3125
Мужской	15	53.57143	23.4375
NA			
Женский	2	33.33333	3.1250
Мужской	4	66.66667	6.2500
Всего			
Всего	64	NA	100.0000

Рисунок 8.88. Таблица *gt* с группировкой по столбцу *diagnosis*

Добавление нового столбца с названием строки в рамках группировки приведет к тому, что этот столбец также станет отдельным блоком.

```
data %>% gt(groupname_col = "diagnosis", rowname_col = "sex")
```

	n	percent_diag	percent_overall
lymphoid			
Женский	15	50.00000	23.4375
Мужской	15	50.00000	23.4375
myeloid			
Женский	13	46.42857	20.3125
Мужской	15	53.57143	23.4375
NA			
Женский	2	33.33333	3.1250
Мужской	4	66.66667	6.2500
Всего			
Всего	64	NA	100.0000

Рисунок 8.89. Таблица *gt* с группировкой по столбцу *diagnosis* и строкам *sex*

К таким сгруппированным таблицам можно применять агрегирующие функции. Для наглядности можно исключить строку итогов и отсутствующий диагноз (значение NA) из набора данных.

```
data %>%
  filter(diagnosis != "Всего") %>%
  gt(groupname_col = "diagnosis", rowname_col = "sex") %>%
  summary_rows(
    groups = c("Lymphoid", "myeloid"),
    columns = n,
    fns = list(
      `Всего` = ~ sum(., na.rm = TRUE)
    ),
    side = "bottom"
  )
```

	n	percent_diag	percent_overall
lymphoid			
Женский	15	50.00000	23.4375
Мужской	15	50.00000	23.4375
Всего	30	—	—
myeloid			
Женский	13	46.42857	20.3125
Мужской	15	53.57143	23.4375
Всего	28	—	—

Рисунок 8.90. Таблица *gt* с группировкой по столбцу *diagnosis* и строкам *sex*, а также расчетом суммарных данных по каждому типу диагноза

Но если необходимо сгруппировать строки только визуально, можно оставить группирующий столбец в качестве столбца (аргумент *row_group_as_column = TRUE*).

```
data %>% gt(groupname_col = "diagnosis", row_group_as_column = TRUE)
```

	sex	n	percent_diag	percent_overall
lymphoid	Женский	15	50.00000	23.4375
	Мужской	15	50.00000	23.4375
myeloid	Женский	13	46.42857	20.3125
	Мужской	15	53.57143	23.4375
NA	Женский	2	33.33333	3.1250
	Мужской	4	66.66667	6.2500
Всего	Всего	64	NA	100.0000

Рисунок 8.91. Таблица *gt* с группировкой по столбцу *diagnosis* с сохранением последнего в качестве отдельного столбца

Итоговая таблица со всеми ранее описанными настройками представлена на **рисунке 8.92**.

Типы диагнозов

Вторая строка заголовка

Пол	Случаев	Процент	
		в группе ¹	всего ²
lymphoid	Женский	15	50.00%
	Мужской	15	50.00%
myeloid	Женский	13	46.43%
	Мужской	15	53.57%
NA	Женский	2	33.33%
	Мужской	4	66.67%
Всего	Всего	64	100.00%

¹ Доля пациента в рамках диагноза

² Доля пациентов в рамках исследования

На основании данных из набора *cytomegalovirus*

Рисунок 8.92. Таблица *gt* с графическими настройками, описанными ранее

В результате группировки данных получилась таблица, практически идентичная предыдущей. Однако, стоит обратить внимание, что столбец с диагнозом утратил заголовок и стал отображаться обычным шрифтом, а не курсивом. Это связано с тем, что после группировки столбец с диагнозом стал не просто столбцом в теле таблицы (*table body*), а "ярлыком группы строк" (*row group label*). Соответственно, для его форматирования нужно использовать отдельные методы, отличные от стандартных стилей для столбцов.

```
gt <- gt %>%
```

```

# Курсив и форматирование для названий групп.
tab_style(
  style = cell_text(style = "italic"),
  locations = cells_row_groups()
) %>%
text_transform(
  locations = cells_row_groups(),
  fn = function(x) {
    lapply(x, function(x) {
      x <- str_to_title(x)
    })
  }
) %>%
# Замена NA в названиях групп.
text_transform(
  locations = cells_row_groups(groups = "NA"),
  fn = function(x){ "Неуточненный" }
) %>%
# Замена Всего в столбце Пол.
text_transform(
  locations = cells_body(columns = sex, rows = sex == "Всего"),
  fn = function(x){ "" }
) %>%
# Выделение строки итогов.
tab_style(
  style = cell_text(font = "Arial", style = "normal", weight = "bold"),
  locations = cells_row_groups(groups = "Всего")
) %>%
# Добавление заголовка с столбцу с названиями групп.
tab_stubhead(label = "Диагноз") %>%
tab_style(
  style = cell_text(font = "Arial", style = "normal", weight = "bold",
                    v_align = "middle"),
  locations = cells_stubhead()
)
gt

```

Типы диагнозов

Вторая строка заголовка

Диагноз	Пол	Случаев	Процент	
			в группе ¹	всего ²
<i>Lymphoid</i>	Женский	15	50.00%	23.44%
	Мужской	15	50.00%	23.44%
<i>Myeloid</i>	Женский	13	46.43%	20.31%
	Мужской	15	53.57%	23.44%
<i>Неуточненный</i>	Женский	2	33.33%	3.12%
	Мужской	4	66.67%	6.25%
Всего		64		100.00%

¹ Доля пациента в рамках диагноза

² Доля пациентов в рамках исследования

На основании данных из набора *cytomegalovirus*

Рисунок 8.93. Таблица *gt* с графическими настройками, описанными ранее и индивидуальной настройкой столбцов и строк

Выделение границ

Границы для таблицы в целом можно определить через функцию *tab_options()*, а также с помощью индивидуального оформления ячеек с использованием функции *tab_style()*. Для *tab_options* параметры внешних границ определяются для блоков таблицы с помощью параметров:

- **.border.top.style* - стиль линии. Возможные значения: *solid* - линия (по умолчанию), *double* - двойная, *dashed* - черточки, *dotted* - точки, *hidden* - скрыть границу;
- **.border.top.width* - ширина границы в пикселях;
- **.border.top.color* - цвет границы.

Для некоторых отдельных блоков, например `table_body` и `column_labels` можно дополнительно к внешней границе border задать еще две группы параметров, которые определяют границы между ячейками: `vlines` - для вертикальных линий и `hlines` - для горизонтальных линий. Для наглядности далее представлен программный код для окрашивания некоторых линий таблицы разными цветами.

```
gt_colored <- gt %>%
  tab_options(
    # Скрыть верхние и нижние границы для таблицы.
    table.border.top.style = "hidden",
    table.border.bottom.style = "hidden",
    # Настройка границы для заголовка.
    heading.border.bottom.style = "solid",
    heading.border.bottom.width = 2,
    heading.border.bottom.color = "green",
    # Настройка линий в заголовках столбцов.
    column_labels.border.top.style = "solid",
    column_labels.border.top.width = 2,
    column_labels.border.top.color = "red",
    column_labels.border.bottom.style = "solid",
    column_labels.border.bottom.width = 2,
    column_labels.border.bottom.color = "red",
    # Настройка верхних и нижних границ для тела таблицы.
    table_body.border.top.style = "solid",
    table_body.border.top.color = "blue",
    table_body.border.top.width = 2,
    table_body.border.bottom.style = "solid",
    table_body.border.bottom.color = "black",
    table_body.border.bottom.width = 2,
    # Настройка вертикальных и горизонтальных линий в теле таблицы.
    table_body.hlines.style = "solid",
    table_body.hlines.width = 1,
    table_body.hlines.color = "orange",
    table_body.vlines.style = "hidden",
    # Группы строк.
    row_group.border.top.width = 1,
    row_group.border.bottom.width = 1
  )
gt_colored
```

Типы диагнозов

Вторая строка заголовка

Диагноз	Пол	Случаев	Процент	
			в группе ¹	всего ²
<i>Lymphoid</i>	Женский	15	50.00%	23.44%
	Мужской	15	50.00%	23.44%
<i>Myeloid</i>	Женский	13	46.43%	20.31%
	Мужской	15	53.57%	23.44%
<i>Неуточненный</i>	Женский	2	33.33%	3.12%
	Мужской	4	66.67%	6.25%
Всего		64		100.00%

¹ Доля пациента в рамках диагноза

² Доля пациентов в рамках исследования

На основании данных из набора *cytomegalovirus*

Рисунок 8.94. Таблица *gt* с настройкой цветных линий

Непосредственно для финального вывода таблицы лучше использовать черный цвет и оттенки серого.

Типы диагнозов

Вторая строка заголовка

Диагноз	Пол	Случаев	Процент	
			в группе ¹	всего ²
<i>Lymphoid</i>	Женский	15	50.00%	23.44%
	Мужской	15	50.00%	23.44%
<i>Myeloid</i>	Женский	13	46.43%	20.31%
	Мужской	15	53.57%	23.44%
<i>Неуточненный</i>	Женский	2	33.33%	3.12%
	Мужской	4	66.67%	6.25%
Всего		64		100.00%

¹ Доля пациента в рамках диагноза

² Доля пациентов в рамках исследования

На основании данных из набора *cytomegalovirus*

Рисунок 8.95. Таблица *gt* с настройкой цвета линий (черный и оттенки серого)

Для более детальных линий следует использовать функцию *tab_style()* и вспомогательную функцию для рисования границ *cell_borders()* с параметрами:

- *sides* - с каких сторон рисуются границы. Возможные значения *left*, *right*, *top*, *bottom*;
- *color* - цвет границы;
- *style* - стиль линии;
- *weight* - толщина линии. Задается в пикселях *px(1)*.

```
gt <- gt %>%
  # Граница для столбца Случаев.
  tab_style(
    locations = list(
      cells_body(columns = c("n")),
      cells_column_labels(columns = c("n"))),
    style = cell_borders(sides = 'right', color = 'black', weight = px(2))
  ) %>%
```

```

# Удаление горизонтальных линий у ячеек с диагнозами.
tab_style(
  locations = cells_body(columns = "diagnosis", rows = seq(1, 5, by=2)),
  style = cell_borders(sides = 'bottom', color = "white", weight = 1)
) %>
# Граница для итоговой строки.
tab_style(
  locations = list(
    cells_body( rows = diagnosis == "Всего" ),
    cells_row_groups(groups = "Всего" )
  ),
  style = cell_borders(sides = 'top', color = 'black', weight = px(2))
)
)
gt

```

Типы диагнозов

Вторая строка заголовка

Диагноз	Пол	Случаев	Процент	
			в группе ¹	всего ²
<i>Lymphoid</i>	Женский	15	50.00%	23.44%
	Мужской	15	50.00%	23.44%
<i>Myeloid</i>	Женский	13	46.43%	20.31%
	Мужской	15	53.57%	23.44%
<i>Неуточненный</i>	Женский	2	33.33%	3.12%
	Мужской	4	66.67%	6.25%
Всего		64		100.00%

¹ Доля пациента в рамках диагноза

² Доля пациентов в рамках исследования

На основании данных из набора *cytomegalovirus*

Рисунок 8.96. Таблица *gt* с дополнительной настройкой линий

Окрашивание элементов

Цвет элементов таблицы можно также определить двумя способами: через функцию `tab_options()` для таблицы в целом, и индивидуально с помощью `tab_style()`. Для определения цвета заливки ячеек используется вспомогательная функция `cell_fill()`.

```
gt <- gt %>%
  # Заголовки столбцов для всей таблицы.
  tab_options(
    column_labels.background.color = 'grey95'
  ) %>%
  # Выделение строк, в которых проценты в группе больше 50.
  tab_style(
    locations = cells_body(
      columns = c("sex", "n", "percent_diag", "percent_overall"),
      rows = percent_diag > 50),
    style = cell_fill(color = '#ffa50040')
  ) %>%
  # Выделение текста, где процент в группе больше 50.
  tab_style(
    locations = cells_body(columns = "percent_diag", rows = percent_diag > 50),
    style = cell_text( color = 'red', weight = 'bold')
  )
)
gt
```

Типы диагнозов

Вторая строка заголовка

Диагноз	Пол	Случаев	Процент	
			в группе ¹	всего ²
<i>Lymphoid</i>	Женский	15	50.00%	23.44%
	Мужской	15	50.00%	23.44%
<i>Myeloid</i>	Женский	13	46.43%	20.31%
	Мужской	15	53.57%	23.44%
<i>Неуточненный</i>	Женский	2	33.33%	3.12%
	Мужской	4	66.67%	6.25%
Всего		64		100.00%

¹ Доля пациента в рамках диагноза

² Доля пациентов в рамках исследования

На основании данных из набора *cytomegalovirus*

Рисунок 8.97. Таблица `gt` с цветовым оформлением ячеек по условию

Создание собственной темы

В отличие от пакета *flextable*, в пакете *gt* большинство настроек по оформлению таблицы можно установить с помощью функции *tab_options()*, поэтому в качестве сохранения стиля оформления таблиц в собственную функцию можно включить только ее.

```
format_my_gt_table <- function(data) {  
  data %>% gt() %>% tab_options(  
    # Убираем верхние и нижние границы для таблицы.  
    table.border.top.style = "hidden",  
    table.border.bottom.style = "hidden",  
    # Установить границы для заголовка.  
    heading.border.bottom.style = "solid",  
    heading.border.bottom.width = 2,  
    heading.border.bottom.color = "black",  
    # Установить линии в заголовках столбцов.  
    column_labels.border.top.style = "solid",  
    column_labels.border.top.width = 2,  
    column_labels.border.top.color = "black",  
    column_labels.border.bottom.style = "solid",  
    column_labels.border.bottom.width = 2,  
    column_labels.border.bottom.color = "black",  
    # Установить верхние и нижние границы для тела таблицы.  
    table_body.border.top.style = "solid",  
    table_body.border.top.color = "black",  
    table_body.border.top.width = 2,  
    table_body.border.bottom.style = "solid",  
    table_body.border.bottom.color = "black",  
    table_body.border.bottom.width = 2,  
    # Установить вертикальные и горизонтальные линии в теле таблицы  
    table_body.hlines.style = "solid",  
    table_body.hlines.width = 1,  
    table_body.hlines.color = "gray90",  
    table_body.vlines.style = "single",  
    # Группы строк.  
    row_group.border.top.width = 1,  
    row_group.border.bottom.width = 1,  
    stub_row_group.border.width = 1,  
    stub_row_group.border.color = "white",  
  )  
}
```

В пакет *gt* включены и шесть собственных тем оформления, которые можно активировать с помощью функции *opt_stylize()*. Данная функция принимает два параметра: *style* - номер темы от 1 до 6 и *color* - базовый цвет, который используется для оформления. По умолчанию при создании таблиц *gt* используется тема под номером 1, однако и для нее можно поменять базовый цвет.

```
data %>% gt() %>% opt_stylize(style = 1, color = 'blue')
```

diagnosis	sex	n	percent_diag	percent_overall
lymphoid	Женский	15	50.00000	23.4375
lymphoid	Мужской	15	50.00000	23.4375
myeloid	Женский	13	46.42857	20.3125
myeloid	Мужской	15	53.57143	23.4375
NA	Женский	2	33.33333	3.1250
NA	Мужской	4	66.66667	6.2500
Всего	Всего	64	NA	100.0000

Рисунок 8.98. Таблица *gt* с использованием цветовой темы №1 и базовым голубым цветом

```
data %>% gt() %>% opt_stylize(style = 6, color = 'green')
```

diagnosis	sex	n	percent_diag	percent_overall
lymphoid	Женский	15	50.00000	23.4375
lymphoid	Мужской	15	50.00000	23.4375
myeloid	Женский	13	46.42857	20.3125
myeloid	Мужской	15	53.57143	23.4375
NA	Женский	2	33.33333	3.1250
NA	Мужской	4	66.66667	6.2500
Всего	Всего	64	NA	100.0000

Рисунок 8.99. Таблица *gt* с использованием цветовой темы №6 и базовым зеленым цветом

Еще больше дополнительных тем и возможностей включены в пакет *gtExtras*.

```
data %>% gt() %>% gt_theme_excel(color = "Lightgrey")
```

diagnosis	sex	n	percent_diag	percent_overall
lymphoid	Женский	15	50.00000	23.4375
lymphoid	Мужской	15	50.00000	23.4375
myeloid	Женский	13	46.42857	20.3125
myeloid	Мужской	15	53.57143	23.4375
NA	Женский	2	33.33333	3.1250
NA	Мужской	4	66.66667	6.2500
Всего	Всего	64	NA	100.0000

Рисунок 8.100. Таблица *gt* с использованием цветовой темы Excel и базовым серым цветом

Дополнительные возможности с *gtExtra*

Кроме набора тем, пакет *gtExtra* позволяет также добавлять в таблицы изображения, иконки, графики и даже интерактивные элементы.

Например, можно сделать заливку значений с помощью тепловой карты.

```
cytomegalovirus %>% head(10) %>%
  select(ID, diagnosis, TNC.dose, CD34.dose, CD3.dose) %>%
  gt() %>% data_color(columns = TNC.dose:CD3.dose, colors = c("white", "red"))
```

ID	diagnosis	TNC.dose	CD34.dose	CD3.dose
1	acute myeloid leukemia	18.31	2.29	3.21
2	non-Hodgkin lymphoma	4.26	2.04	NA
3	non-Hodgkin lymphoma	8.09	6.97	2.19
4	Hodgkin lymphoma	21.02	6.09	4.87
5	acute lymphoblastic leukemia	14.70	2.36	6.55
6	myelofibrosis	4.29	6.91	2.53
7	acute myeloid leukemia	7.96	3.66	3.66
8	acute myeloid leukemia	15.63	3.90	7.27
9	multiple myelomas	6.86	7.00	2.59
10	chronic lymphocytic leukemia	7.54	2.52	2.52

Рисунок 8.101. Таблица *gt* с использованием тепловой карты

Или добавить в отдельный столбец график, иллюстрирующий характер сгруппированных данных.

```
data <- cytomegalovirus %>%
  group_by(diagnosis) %>%
  summarise(
    mean = mean(time.to.transplant),
    sd = sd(time.to.transplant),
    time.to.transplant.data = list(time.to.transplant),
    .groups = "drop"
  )
gt_plot <- data %>% arrange(desc(diagnosis)) %>% head() %>% gt() %>%
  gt_plt_dist(time.to.transplant.data, type = "density", line_color = "blue",
  fill_color = "red") %>%
  fmt_number(columns = mean:sd, decimals = 1)
gt_plot
```

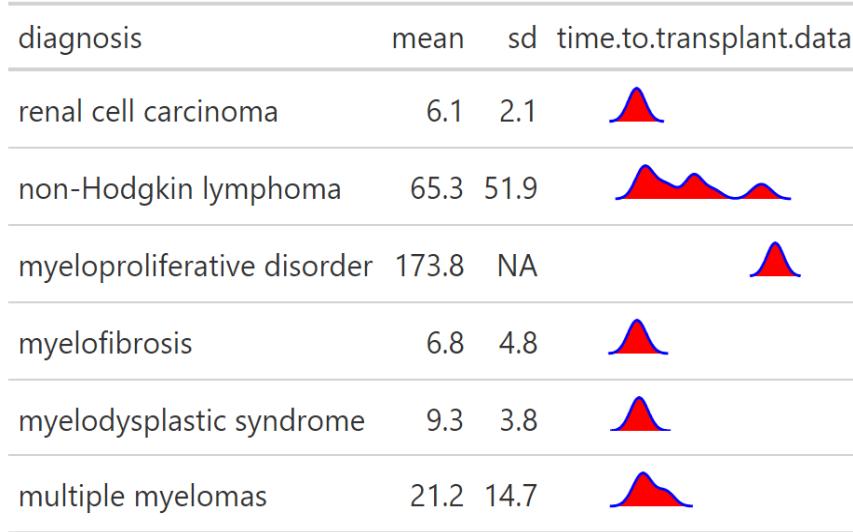


Рисунок 8.102. Таблица *gt* с использованием графиков распределения данных

Сохранение таблицы

Для сохранения готовой таблицы *gt* в файл используется функция *gtsave()*. С ее помощью можно сохранить таблицы в форматах HTML, DOCX, RTF, TEX, LTX и PNG.

```
gt %>% gtsave("gt.html")
gt %>% gtsave("gt.docx")
gt %>% gtsave("gt.png")
```

Если при сохранении в HTML таблица предстает идентичной ее отображению в RStudio, то при сохранении в DOCX часть оформления может быть потеряна.

В некоторых случаях, например при сохранении таблицы в виде изображения для печати, размер получаемого изображения может показаться слишком маленьким. В

в этом случае необходимо воспользоваться функцией `gtsave_extra()` из пакета `gtExtra`, которая позволяет задать требуемый уровень масштабирования.

```
gt %>% gtsave_extra("gt.png", zoom = 3, expand = 5)
```

8.2.4. Создание отчетов в формате Word

Логика статистического анализа предполагает взаимосвязь между результатами расчетов. Поэтому оптимальным решением будет объединение всех расчетов и их выводов в одном документе, обеспечивая целостность и удобство анализа. Пакеты в языке R предоставляют множество возможностей для формирования таких отчетов. Если рассматривать создание документов в формате `.docx`, то наиболее популярным в настоящее время является пакет `officer`. Он позволяет создавать документы Word и PowerPoint прямо при выполнении скрипта R. Пример работы с пакетом `officer` также представлен в конце [раздела 6.7.3.](#), который посвящен концепции воспроизводимых исследований.

Общая логика работы с пакетом следующая:

1. Создание виртуального документа в памяти: `doc <- read_docx()`
2. Добавление нужных элементов в документ: `body_add_par(doc, "Это пример обычного текста")`
3. Сохранение документа: `print(doc, target = "data/sample_file.docx")`

Пакет позволяет сохранять в Word документе текст, таблицы, изображения и графики. При этом допускается задание стилей оформления, иерархические заголовки, автоматическое создание оглавления. Набор стандартных стилей ограничен, но их можно переопределить в документе, который будет использоваться в качестве шаблона. Далее представлен пример создания отчета.

В качестве первого шага необходимо создать документ в памяти, с которым будет производиться работа. Можно создать пустой документ или воспользоваться существующим шаблоном, который позволяет переопределить стандартные стили, задать параметры страницы и другие параметры.

```
doc <- read_docx(path = "templates/template.docx")
```

После этого в созданный документ можно добавлять различные объекты. Для этого предусмотрены следующие функции:

- `body_add_blocks()` - блок;
- `body_add_break()` - разрыв страницы;
- `body_add_caption()` - подпись;
- `body_add_docx()` - содержимое документа docx;
- `body_add_fpar()` - абзац форматированного текста;

- `body_add_gg()` - график `ggplot2`;
- `body_add_img()` - изображение;
- `body_add_par()` - абзац текста без форматирования;
- `body_add_plot()` - график;
- `body_add_table()` - таблица;
- `body_add_toc()` - оглавления.

С помощью дополнительных пакетов можно расширить доступный набор функций: например, добавлять таблицы `flextable` и `gt`. Для некоторых функций можно определить стиль содержимого: для текста задать иерархическую структуру заголовков или оформить подписи к объектам, для таблиц - задать их внешний вид.

Стандартный набор предусмотренных стилей следующий:

- Для текста:
 - Normal;
 - heading 1;
 - heading 2;
 - heading 3;
 - centered.
- Для оглавления:
 - toc 1;
 - toc 2.
- Для подписей объектов:
 - Image Caption;
 - Table Caption;
 - Balloon Text;
 - graphic title;
 - table title.
- Для таблиц:
 - Normal Table;
 - table_template;
 - Light List Accent 2;
 - Table Professional.

```
# Создание нового документа в памяти.
doc <- read_docx()
# Запись в документ в него заголовка.
doc <- doc %>% body_add_par("Заголовок документа", style = "heading 1")
# Вставка оглавления.
doc <- body_add_toc(doc, style = "toc 1")
# После оглавления вставка разрыва страницы.
doc <- body_add_break(doc)
# Сохранение результата в файл на диске.
print(doc, target = "data/officer_report.docx")
```

Сохранять файл на диск можно по мере заполнения документа, чтобы контролировать процесс формирования документа.

```

# Добавление текста.
doc <- body_add_par(doc, "Текст", style = "heading 2")
doc <- body_add_par(doc, "Это пример обычного текста со стандартным оформлением")
doc <- body_add_par(doc, "Таблицы", style = "heading 2")
# Добавление таблицы.
doc <- body_add_par(doc, "Простая таблица", style = "Table Caption")
doc <- body_add_table(doc, data)
# Добавление таблицы со стилем.
doc <- body_add_par(doc, "Таблица со стилем Table Professional", style = "Table
Caption")
doc <- body_add_table(doc, data, style = "Table Professional")
# Добавление таблицы со стилем.
doc <- body_add_par(doc, "Таблица со стилем table_template", style = "Table
Caption")
doc <- body_add_table(doc, data, style = "table_template")

```

1.2. Таблицы

Простая таблица

diagnosis	sex	n	percent_diag	percent_overall
lymphoid	Женский	15	50.00000	23.4375
lymphoid	Мужской	15	50.00000	23.4375
myeloid	Женский	13	46.42857	20.3125
myeloid	Мужской	15	53.57143	23.4375
NA	Женский	2	33.33333	3.1250
NA	Мужской	4	66.66667	6.2500
Всего	Всего	64	NA	100.0000

Таблица со стилем Table Professional

diagnosis	sex	n	percent_diag	percent_overall
lymphoid	Женский	15	50.00000	23.4375
lymphoid	Мужской	15	50.00000	23.4375
myeloid	Женский	13	46.42857	20.3125
myeloid	Мужской	15	53.57143	23.4375
NA	Женский	2	33.33333	3.1250
NA	Мужской	4	66.66667	6.2500
Всего	Всего	64	NA	100.0000

Таблица со стилем table_template

diagnosis	sex	n	percent_diag	percent_overall
lymphoid	Женский	15	50.00000	23.4375
lymphoid	Мужской	15	50.00000	23.4375
myeloid	Женский	13	46.42857	20.3125
myeloid	Мужской	15	53.57143	23.4375
NA	Женский	2	33.33333	3.1250
NA	Мужской	4	66.66667	6.2500
Всего	Всего	64	NA	100.0000

Рисунок 8.103. Таблицы с различными стилями, сохраненные в Word

Как можно заметить, стили оформления таблиц достаточно специфичны и стандартны. Для более изысканного оформления можно использовать таблицы, созданные с помощью пакета *flextable*, которые идеально отображаются в документе DOCX.

```

doc <- body_add_par(doc, "Таблица flextable", style = "heading 3")
doc <- body_add_flextable(doc, ft_full)

```

1.2.1. Таблица flextable

Типы диагнозов

Диагноз	Пол	Случаев	Процент	
			в группе ¹	всего ²
<i>Lymphoid</i>	Женский	15	50.00%	23.44%
	Мужской	15	50.00%	23.44%
<i>Myeloid</i>	Женский	13	46.43%	20.31%
	Мужской	15	53.57%	23.44%
	Женский	2	33.33%	3.12%
	Мужской	4	66.67%	6.25%
Всего		64	100.00%	

На основании данных из набора супотегаловирус

¹Доля пациента в рамках диагноза

²Доля пациентов в рамках исследования

Рисунок 8.104. Таблица *flextable*, сохраненная в Word

Здесь можно вспомнить про функцию *set_table_properties()*, которая определяла ширину таблицы — в ней можно определить сколько места будет занимать таблица в результирующем документе при выводе. Далее представлен код, позволяющий разместить таблицу по всей ширине страницы документа.

```
doc <- body_add_par(doc, "Таблица flextable во всю ширину", style = "heading 3")
doc <- body_add_flextable(doc, ft_full %>% set_table_properties(width = 1, layout = "autofit"))
```

1.2.2. Таблица flextable во всю ширину

Типы диагнозов

Диагноз	Пол	Случаев	Процент	
			в группе ¹	всего ²
<i>Lymphoid</i>	Женский	15	50.00%	23.44%
	Мужской	15	50.00%	23.44%
<i>Myeloid</i>	Женский	13	46.43%	20.31%
	Мужской	15	53.57%	23.44%
	Женский	2	33.33%	3.12%
	Мужской	4	66.67%	6.25%
Всего		64	100.00%	

На основании данных из набора супотегаловирус

¹Доля пациента в рамках диагноза

²Доля пациентов в рамках исследования

Рисунок 8.105. Таблица *flextable*, сохраненная в Word во всю ширину

Таблицы, созданные пакетом *gt*, тоже можно добавлять в документ, но для этого потребуется установить дополнительный пакет *gto*. После этого станет доступна функция *body_add_gt()*. Однако таблицы *gt* не имеют такой идеальной совместимости с пакетом *officer*, поэтому результат в отчете может достаточно сильно отличаться от

того, что представлено при просмотре результатов в RStudio. В этом случае таблицы *gt* необходимо настраивать индивидуально, уже непосредственно для работы с *officer*.

```
doc <- body_add_par(doc, "Таблица gt", style = "Table Caption")
doc <- body_add_gt(doc, gt)
```

1.2.3. Таблица *gt*

Table 1: Типы диагнозов

Вторая строка заголовка

Пол	Случаев	Процент	
		в группе ¹	всего ²
<i>Lymphoid</i>			
Женский	15	50.00%	23.44%
Мужской	15	50.00%	23.44%
<i>Myeloid</i>			
Женский	13	46.43%	20.31%
Мужской	15	53.57%	23.44%
<i>Неуточненный</i>			
Женский	2	33.33%	3.12%
Мужской	4	66.67%	6.25%
Всего			
	64		100.00%

¹Доля пациента в рамках диагноза

²Доля пациентов в рамках исследования

На основании данных из набора *cytomegalovirus*

Рисунок 8.106. Таблица *gt*, сохраненная в Word

Пакет *officer* позволяет добавлять в документ различные типы изображений. Это могут быть как обычные файлы, так и графики, в том числе полученные с помощью пакета *ggplot2*. В отличие от вставки графиков в виде изображений, пакет *mschart* позволяет создавать графики, которые выглядят как стандартные диаграммы MS Office и могут редактироваться непосредственно в документе. Это дает возможность изменять стили, значения и другие параметры графика без необходимости переходить в отдельный редактор.

```
# Добавление файла изображения в документ.
doc <- body_add_par(doc, "Это график из файла", style = "Image Caption")
doc <- body_add_img(doc, src = "data/boxplot.png", width = 5, height = 5)
# Добавление графика из кода.
doc <- body_add_par(doc, "Это график из кода", style = "Image Caption")
doc <- body_add_plot(doc, value = boxplot(age ~ sex, cytomegalovirus, col =
"green"), width = 4, height = 4, res = 300)
# Добавление графика ggplot.
library(ggplot2)
gg <- ggplot(cytomegalovirus %>% mutate(sex = ifelse(sex == 1, "M", "F")),
aes(sex, age)) + geom_boxplot()
doc <- body_add_par(doc, "Это график из ggplot", style = "Image Caption")
doc <- body_add_gg(doc, value = gg, width = 4, height = 4, res = 300)
# Добавление графика MS Office.
library(mschart)
bc <- ms_barchart(data = cytomegalovirus %>% mutate(sex = ifelse(sex == 1, "M",
"F")), x = "sex", y = "age", group = "diagnosis")
```

```
doc <- body_add_par(doc, "Это график MS Office", style = "Image Caption")
doc <- body_add_chart(doc, chart = bc, style = "Normal", width = 6, height = 8)
```

После добавления всех необходимых элементов документ можно сохранить.

```
print(doc, target = "data/officer_report.docx")
```

В результате выполнения всех вышепредставленных функций получился документ, представленный на **рисунке 8.107**.



Рисунок 8.107. Созданный документ в формате DOCX

Для удобства генерации документа, команды можно объединять в пайплайн.

```
doc <- read_docx() %>%
  body_add_par("Заголовок документа", style = "heading 1") %>%
  body_add_toc(Level = 3) %>%
  body_add_break() %>%
  body_add_par("Это пример обычного текста со стандартным оформлением") %>%
  body_add_par("Простая таблица", style = "table title") %>%
  body_add_table(mtcars) %>%
```

```
print(target = "data/officer_report_2.docx")
```

Это краткий обзор функций, доступных в пакете *officer* для работы с MS Word документами. Кроме перечисленного, данный пакет позволяет объединять несколько существующих документов в один, задавать различные параметры страниц для разных секций документа, например, в случаях когда график или таблицу нужно разместить на странице альбомной ориентации. Также с помощью пакета *officer* можно создавать презентации PowerPoint.

8.2.5. Создание отчетов в формате PowerPoint

Кроме создания документов MS Word, пакет *officer* также позволяет создавать презентации MS PowerPoint.

Общая логика работы остается прежней:

1. Создание виртуального документа в памяти: `doc <- read_pptx()`
2. Добавление слайда: `add_slide(doc, layout = "Title and Content", master = "Office Theme")`
3. Добавление элементов на слайд: `ph_with(doc, "Текст с оформлением", ph_location_type(type = "title"))`
4. Сохранение документа: `print(doc, target = "data/officer_report.pptx")`

При добавлении нового слайда в параметре *layout* необходимо указать шаблона слайда, который будет добавлен. Доступны следующие варианты шаблонов слайдов:

- Title Slide - титульный слайд;
- Title and Content - заголовок и содержание;
- Section Header - заголовок раздела;
- Two Content - заголовок и две колонки;
- Comparison - заголовок и две колонки - каждая с заголовком и содержимым;
- Title Only - один заголовок;
- Blank - пустой слайд.

Все элементы добавляются на слайд только одной командой `ph_with(document, value, location)` в которой указывается в какой документ добавляется, какой элемент и на какую позицию. Допустимые варианты позиций на слайде:

- `ph_location_type()`:
 - *body* - основное содержимое;
 - *dt* - дата - левый нижний угол;
 - *ftr* - футер слайда - внизу и по центру;
 - *sldNum* - номер слайда - правый нижний угол;
 - *title* - заголовок слайда.
- `ph_location_fullsize()` - добавляется на весь слайд;
- `ph_location_label()` - расположение по указанной метке;

- `ph_location_left()` - добавляется в левую колонку;
- `ph_location_right()` - добавляется в правую колонку;
- `ph_location()` - свободное расположение элемента;
- `ph_location_template()` - свободное расположение элемента согласно выбранному шаблону.

Далее рассмотрен пример кода для создания презентации.

```
# Создание презентации.
doc <- read_pptx()
# Добавление слайда.
doc <- add_slide(doc, Layout = "Title and Content", master = "Office Theme")
# Работа со слайдом.
doc <- ph_with(doc, value = "Hello world", location = ph_location_type(type =
"title"))
doc <- ph_with(doc, value = "A footer", location = ph_location_type(type =
"ftr"))
doc <- ph_with(doc, value = format(Sys.Date()), location = ph_location_type(type =
"dt"))
doc <- ph_with(doc, value = "slide 1", location = ph_location_type(type =
"sldNum"))
doc <- ph_with(doc, value = head(Letters), location = ph_location_type(type =
"body"))
```

Hello world

- a
- b
- c
- d
- e
- f

2024-09-30 A footer slide 1

Рисунок 8.108. Созданный слайд PowerPoint

Поскольку в рамках команды `ph_with()` можно добавлять только один элемент, при добавлении сложного оформленного текста `ftext` его необходимо объединять в параграфы (`fpar`), а их в свою очередь в блоки (`block_list`).

```
# Добавление текста.
doc <- add_slide(doc, "Title and Content")
doc <- ph_with(doc, "Текст с оформлением", ph_location_type(type =
"title"))
# Создание параграфа.
par1 <- fpar(
  # Первый блок текста.
  ftext("Это жирный и большой текст", prop = fp_text(bold = TRUE, font.size =
40)),
```

```

# Второй блок текста.
ftext("Этот текст красный и поменьше", prop = fp_text(color = "red", font.size
= 20))
)
# Еще один параграф с текстом без оформления.
par2 <- fpar(ftext("А этот текст будет обычным"))
# Объединение параграфов в блок. Второй параграф продублирован.
bl <- block_list(par1, par2, par2)
# Добавление блока текста на слайд.
doc <- ph_with(doc, bl, ph_location_type(type = "body"))

```

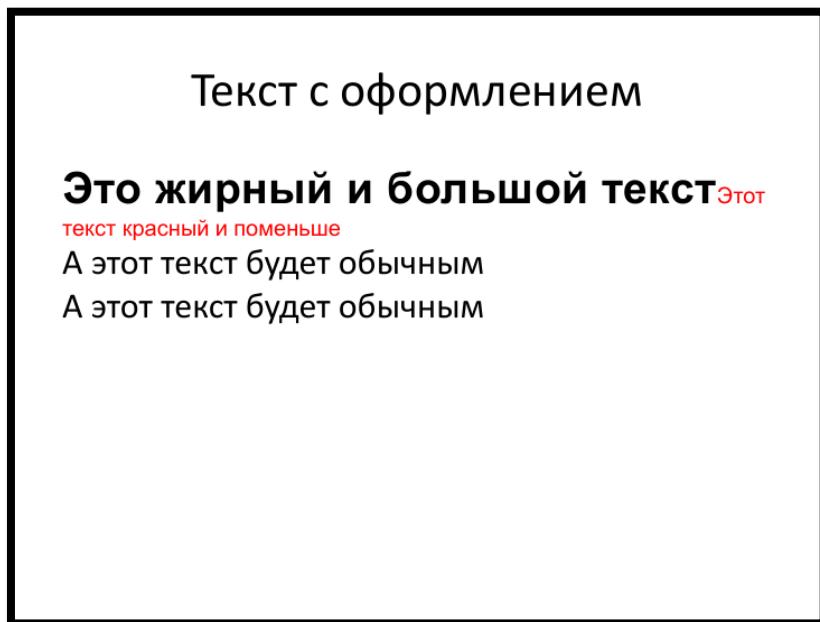


Рисунок 8.109. Созданный слайд PowerPoint с форматированием текста

При добавлении таблицы на слайд она автоматически оформляется встроенными стилями MS Office.

```

doc <- add_slide(doc, "Title and Content")
doc <- ph_with(doc, "Таблица", ph_location_type(type = "title"))
doc <- ph_with(doc, head(data), ph_location_type(type = "body"))

```

Таблица

diagnosis	sex	n	percent_diag	percent_overal l
lymphoid	Женский	15	50.00000	23.4375
lymphoid	Мужской	15	50.00000	23.4375
myeloid	Женский	13	46.42857	20.3125
myeloid	Мужской	15	53.57143	23.4375
NA	Женский	2	33.33333	3.1250
NA	Мужской	4	66.66667	6.2500

Рисунок 8.110. Созданный слайд PowerPoint с табличными данными

При добавлении таблицы *flextable* ее оформление сохраняется.

```
doc <- add_slide(doc, "Title and Content")
doc <- ph_with(doc, "Таблица flextable", ph_location_type(type = "title"))
doc <- ph_with(doc, ft, ph_location_type(type = "body"), alignment = "c",
use_loc_size = TRUE)
```

Вторая строка заголовка

Диагноз	Пол	Случаев	Процент	
			в группе ¹	всего ²
Lymphoid	Женский	15	50.00%	23.44%
	Мужской	15	50.00%	23.44%
Myeloid	Женский	13	46.43%	20.31%
	Мужской	15	53.57%	23.44%
NA	Женский	2	33.33%	3.12%
	Мужской	4	66.67%	6.25%
Всего		64	100.00%	

¹Доля пациента в рамках диагноза
²Доля пациентов в рамках исследования

Рисунок 8.111. Созданный слайд PowerPoint с таблицей *flextable*

Добавляя изображение на слайд, можно указать параметры размеров: задать точные размеры в дюймах или же позволить ему занять всю доступную область.

```

# Добавление слайда с двумя колонками.
doc <- add_slide(doc, "Two Content")
doc <- ph_with(doc, "Картина из файла", ph_Location_type(type = "title"))
# Получение полного пути к картинке.
img <- file.path(getwd(), "data", "boxplot.png")
# Добавление размеров добавим в левую колонку с ручным указанием.
doc <- ph_with(doc, external_img(img, width = 2.78, height = 2.12),
ph_Location_left(), use_loc_size = FALSE)
# Добавление размеров в правую колонку с автоматическим определением.
doc <- ph_with(doc, external_img(img), location = ph_Location_right(),
use_loc_size = TRUE)

```

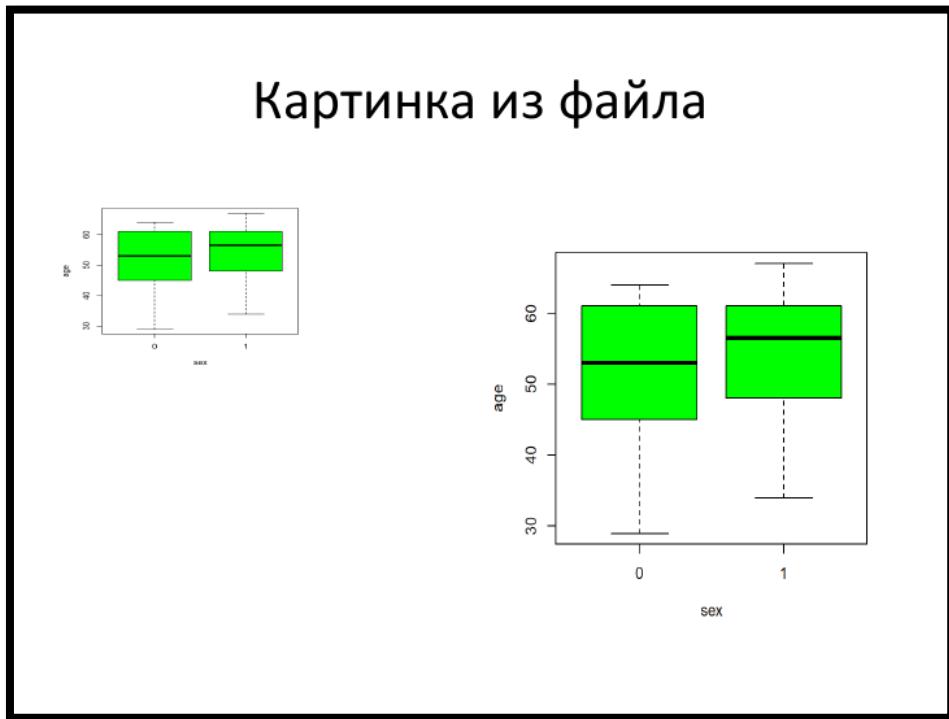


Рисунок 8.112. Созданный слайд PowerPoint со стандартными графиками

Графики *ggplot2* при вставке на слайд заполняют всю предоставленную им область.

```

doc <- add_slide(doc, "Two Content")
doc <- ph_with(doc, "График ggPlot", ph_Location_type(type = "title"))
doc <- ph_with(doc, "График ggPlot занимает всю предоставленную ему область",
location = ph_Location_left())
doc <- ph_with(doc, gg, location = ph_Location_right())

```

График ggplot

- График ggplot занимает всю предоставленную ему область

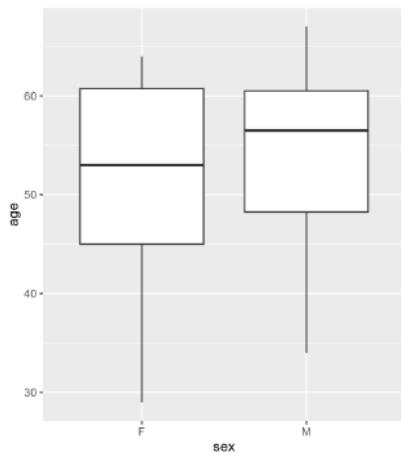


Рисунок 8.113. Созданный слайд PowerPoint с графиком *ggplot2*

Наиболее выгодно графики *ggplot2* выглядят при размещении во весь размер слайда.

```
doc <- add_slide(doc, Layout = "BLank")
doc <- ph_with(doc, gg, ph_location_fullsize() )
```

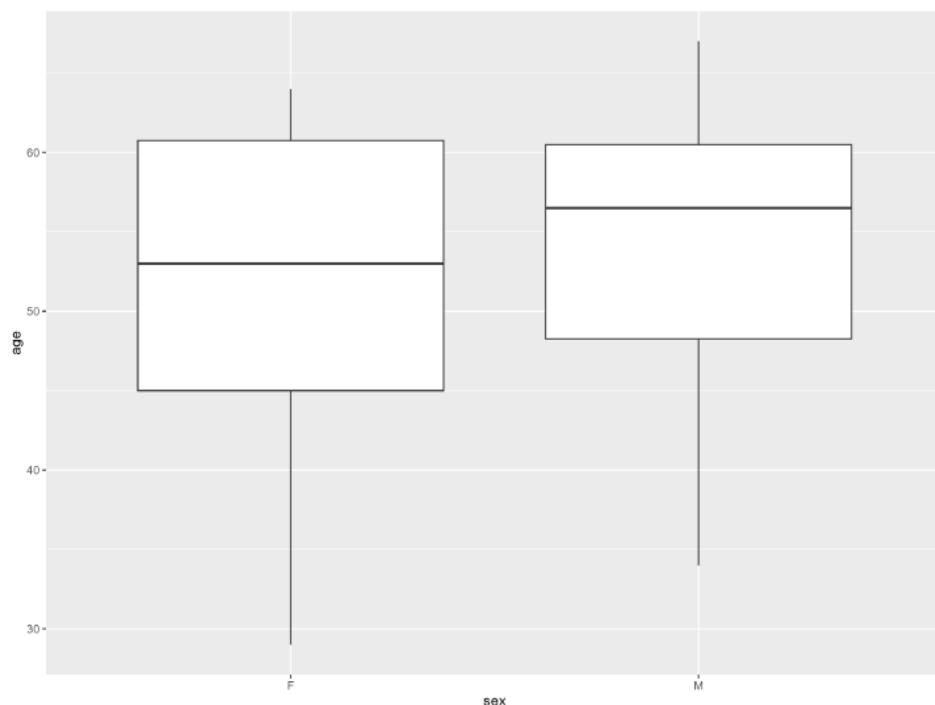


Рисунок 8.114. Созданный слайд PowerPoint с графиком *ggplot2* во весь экран

Графики *mscharts* при добавлении заполняют всю доступную область. При этом они остаются редактируемыми в презентации.

```
doc <- add_slide(doc, Layout = "Two Content")
doc <- ph_with(doc, "График MS Office", ph_location_type(type = "title"))
doc <- ph_with(doc, "Это редактируемый график, созданный средствами MS Office",
              ph_location_left())
doc <- ph_with(doc, bc, ph_location_right())
```

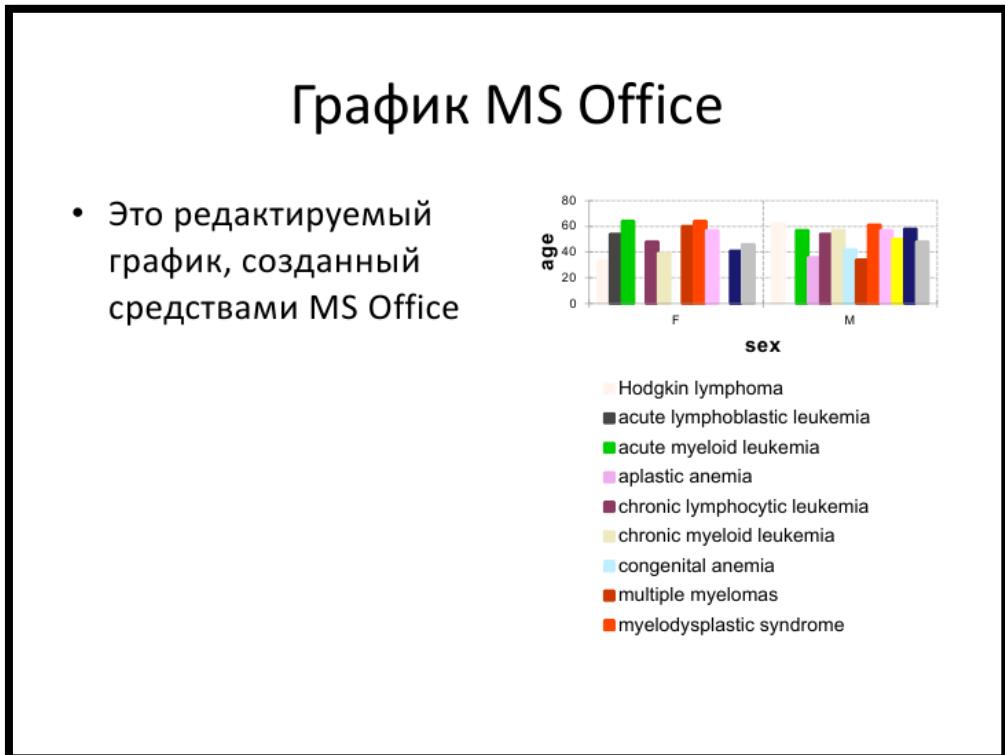


Рисунок 8.115. Созданный слайд PowerPoint с редактируемым графиком *mscharts*

Для сохранения презентации в файл используется команда *print()*.

```
print(doc, target = "data/officer_report.pptx")
```

В данном разделе был представлен краткий обзор функций, доступных в пакете *officer* для работы с презентациями PowerPoint.

8.2.6. Создание отчетов в формате Excel

Пакет *openxlsx* позволяет не просто сохранять таблицы на лист, но и полноценно работать с документом Excel, записывая значения в ячейки, менять оформление, добавлять изображения. Далее рассмотрены некоторые функции для подготовки отчетов в Excel. Перед началом работы необходимо создать рабочую книгу в памяти и добавить на нее лист, с которым будет проводиться работа.

```
# Создание книги.
wb <- createWorkbook()
# Добавление листа.
```

```
sheet = "Данные"
addWorksheet(wb, sheetName = sheet, gridLines = TRUE)
```

Для записи информации в ячейку используется команда `writeData`. Для нее необходимо указать книгу, лист, данные для записи и точную позицию ячейки.

Использование диапазонов упрощает работу с ячейками. Например, для создания заголовка в таблице `data_excel` можно объединить ячейки первой строки от 1 до `ncol(data_excel)` (количество столбцов).

```
# Указание заголовка.
writeData(wb, sheet, "Заголовок для таблицы", startCol = 1, startRow = 1)
# Объединение первых трех ячеек, чтобы записать заголовок.
mergeCells(wb, sheet, cols = 1:ncol(data_excel), rows = 1)
```

Записать саму таблицу `data_excel` в документ можно сразу под заголовком (т.е. начать запись со второй строки Excel, т.к. в первой строке расположен заголовок таблицы). При записи таблицы можно указать, записывать ли имена строк и столбцов.

```
writeData(wb, sheet, data_excel, colNames = TRUE, rowNames = FALSE, startCol = 1,
          startRow = 2)
```

Для диапазонов ячеек можно задавать стили оформления, которые могут определять цвет текста, заливку фона, выравнивание, и так далее. Созданные стили можно использовать повторно и применять в различных местах документа.

```
# Создание стиля для заголовка.
headerStyle <- createStyle(fontSize = 12, fontColour = "black",
                           halign = "center", valign = "center", fgFill =
                           "mistyrose",
                           border = "TopBottomLeftRight", borderColour =
                           "black",
                           textDecoration = "bold", wrapText = TRUE)
# Применение стиля для текста заголовка таблицы.
addStyle(wb, sheet, headerStyle, rows = 1, cols = 1, gridExpand = TRUE)

# Создание стиля для столбцов таблицы.
tableHeaderStyle <- createStyle(fontSize = 12, fontColour = "black",
                           halign = "center", valign = "center", fgFill =
                           "lightcyan2",
                           border = "TopBottomLeftRight", borderColour =
                           "black",
                           textDecoration = "bold", wrapText = TRUE)
# Применение стиля для столбцов таблицы.
addStyle(wb, sheet, tableHeaderStyle, rows = 2, cols = 1:ncol(data_excel),
          gridExpand = TRUE)
```

При вставке изображения важно указывать размеры изображения и ячейку, соответствующую верхнему левому углу изображения.

```
insertImage(wb, sheet, "./data/boxplot.png", width = 6, height = 6, units = "cm",
            startCol = 6, startRow = 1)
```

Аналогично можно добавить график `ggplot2`.

```

gg <- ggplot(cytomegalovirus %>% mutate(sex = ifelse(sex == 1, "M", "F")),
aes(sex, age)) + geom_boxplot()
print(gg)
insertPlot(wb, sheet, width = 6, height = 6, units = "cm", xy= c("F", 1), fileType
= "png")

```

Сохраним документ и получим следующий результат.

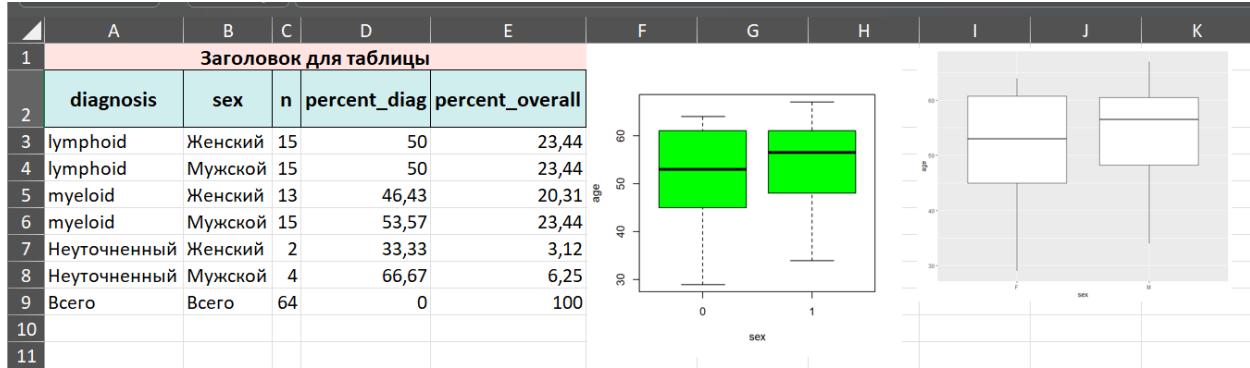


Рисунок 8.116. Созданный отчет в Excel с таблицей и графиком

Кроме статического, можно задавать и динамическое форматирование ячеек с помощью функции *conditionalFormatting()*, основанное на анализе их значений. Далее будут рассмотрены некоторые примеры динамического форматирования. Для демонстрации следует создать новый лист и вставить в него набор данных *cytomegalovirus* (набор данных рассматривался ранее в разделе 2.6.1.).

```

sheet2 = "cytomegalovirus"
addWorksheet(wb, sheet2, gridLines = TRUE)
writeData(wb, sheet2, cytomegalovirus, colNames = TRUE, rowNames = FALSE,
na.string = "")
# Выделение ячеек, где пол пациента равен Male (соответствует sex == 1).
conditionalFormatting(wb, sheet2,
style = createStyle(bgFill = "salmon"),
cols = 3, rows = 1:(1+nrow(cytomegalovirus)),
type = "expression",
rule = "==1")
# Применение градиентного окрашивания для столбца с возрастом.
conditionalFormatting(wb, sheet2,
style = c("white", "red"),
cols = 2, rows = 1:(1+nrow(cytomegalovirus)),
type = "colourScale",
rule = c(29, 70))
# Выделение одинаковых значений в столбце с временем от постановки диагноза до трансплантации.
conditionalFormatting(wb, sheet2,
style = createStyle(bgFill = "gold"),
cols = 7, rows = 1:(1+nrow(cytomegalovirus)),
type = "duplicates")
# Подсветка top 10 самых больших значений времени от постановки диагноза до трансплантации.
conditionalFormatting(wb, sheet2,
style = createStyle(bgFill = "cyan"),
cols = 7, rows = 1:(1+nrow(cytomegalovirus)),
type = "topN",
n = 10)

```

```

rank = 10, percent = TRUE)
# Выделение диагнозов, которые начинаются со слова Hodgkin.
conditionalFormatting(wb, sheet2,
style = createStyle(bgFill = "green"),
cols = 5, rows = 3:(3+nrow(cytomegalovirus)),
type = "beginsWith",
rule = "Hodgkin")

```

В результате выполнения вышепредставленного кода представлены на **рисунке 8.117**.

	A	B	C	D	E	F	G
1	ID	age	sex	race	diagnosis	diagnosis.type	time.to.trans
2		1	61	1	0 acute myeloi	1	5,16
3		2	62	1	1 non-Hodgkin	0	79,05
4		3	63	0	1 non-Hodgkin	0	35,58
5		4	33	0	1 Hodgkin lym	0	33,02
6		5	54	0	1 acute lympho	0	11,4
7		6	55	1	1 myelofibrosi	1	2,43
8		7	67	1	1 acute myeloi	1	9,59
9		8	51	1	1 acute myeloi	1	
10		9	44	0	0 multiple mye	0	43,43
11		10	59	1	1 chronic lymp	0	92,71
12		11	45	1	1 multiple mye	0	39
13		12	57	1	1 acute myeloi	1	17,84
14		13	52	0	1 myelodyspla	1	4,53
15		14	38	0	1 multiple mye	0	21,32
16		15	35	1	1 myelodyspla	1	16,33
17		16	61	0	1 non-Hodgkin	0	162,4
18		17	62	0	1 acute myeloi	1	13,7
19		18	45	0	1 myelodyspla	1	8,71

Рисунок 8.117. Созданный отчет в Excel с форматированием цветом по условию

Из дополнительных функций присутствует возможность защитить паролем как отдельный лист, так и всю книгу целиком.

```

protectWorksheet(wb, sheet, password = "123")
protectWorkbook(wb, password = "123")

```

После завершения работы нужно сохранить документ на диск.

```
saveWorkbook(wb, "data/openxlsx_report.xlsx", overwrite = TRUE)
```

8.2.7. Создание отчетов в формате Rmarkdown

Ранее уже были рассмотрены различные возможности формирования отчетов с помощью пакетов *officer* и *openxlsx*. Однако в экосистеме R есть возможность

создавать интерактивные отчеты на основе технологии Rmarkdown, которая позволяет объединять в одном документе оформление текста с помощью языка разметки Markdown и интерактивного кода на R. Этот код выполняется в момент формирования отчета и его результат вставляется в отчет. После вычисления из этого файла можно получить документы различных форматов. Преимуществом данного подхода является то, что в одном документе можно хранить и сам код, и результат его вычислений, который виден прямо в документе. Ранее эта концепция обсуждалась в **разделе 6.6.3.**, который посвящен воспроизведимым исследованиям.

Описание формата Rmarkdown

Файлы формата Rmarkdown сохраняются с расширением *.Rmd и позволяют в дальнейшем генерировать из них отчеты в форматах Word, PowerPoint, Pdf, Html, а также интерактивные web-страницы. Для того, чтобы начать работу с RMarkdown необходимо установить пакет *rmarkdown*.

```
install.packages("rmarkdown")
```

Создать файлы Rmarkdown в программе Rstudio (**см. раздел 2.1.1.**) можно с помощью пункта меню File -> New File -> R markdown (**рисунок 8.118.**).

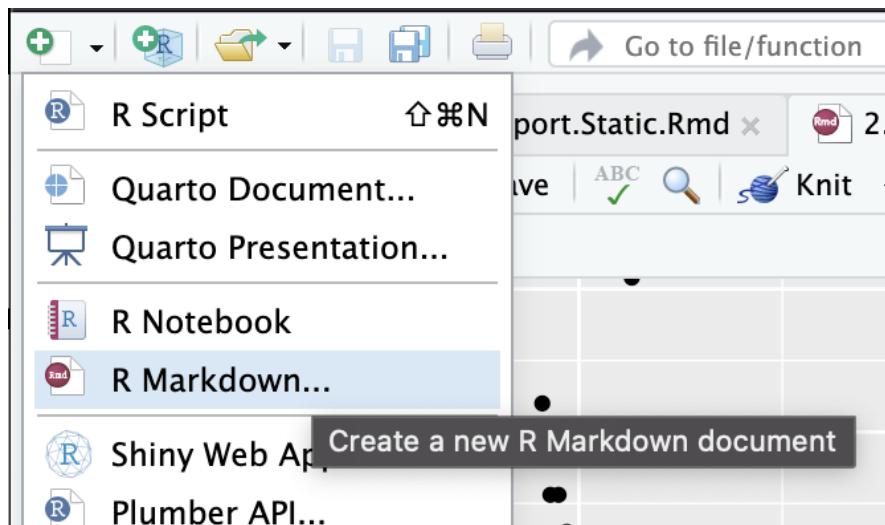


Рисунок 8.118. Создание файла Rmarkdown в программе Rstudio

Каждый такой файл включает два раздела - преамбулу (в разметке Yaml) и содержимое (в разметке Markdown). В преамбуле описываются общие параметры будущего отчетного документа. В содержимом описывается непосредственно содержимое будущего отчета (текст, программный код, изображения и пр.).

Пример преамбулы:

```
---
```

```
title: "Пример отчета с использованием RMarkdown"
author: "Иван Трушин"
```

```
date: "2022-01-03"
output:
  ioslides_presentation: default
  word_document:
    toc: yes
    df_print: kable
    reference_doc: templates/template.docx
  html_document:
    toc: yes
    df_print: kable
  powerpoint_presentation:
    toc: yes
    df_print: kable
    reference_doc: templates/template.pptx
always_allow_html: yes
---
```

Обязательные поля преамбулы:

- *title* - название документа;
- *author* - автор документа;
- *date* - дата создания документа.

Также обязательным должен быть как минимум один элемент *output*, который определяет формат результирующего документа. Таких элементов может быть несколько. В данном примере на основе документа Rmarkdown будет генерировано три типа документов: отчет Word, отчет Html, презентация PowerPoint.

Список документов в которые можно скомпилировать (т.е. “конвертировать”) документ Rmarkdown следующий:

- context_document;
- github_document;
- html_document;
- latex_document;
- md_document;
- odt_document;
- pdf_document;
- rtf_document;
- word_document;
- beamer_presentation;
- ioslides_presentation;
- powerpoint_presentation;
- slidy_presentation.

Для документов Word и PowerPoint можно задать шаблон оформления со стилями с помощью поля *reference_doc*. Далее представлены различные способы описания содержимого документа Rmarkdown.

Статическое содержимое

Обычное статическое содержимое можно описывать в формате Markdown. Примеры разметки текста в формате Markdown представлены далее.

Главный заголовок документа

Текст (для заголовка второго уровня используется два символа решетки ##)

Абзац простого текста. Простой текст можно писать так как он есть.

При необходимости можно разбивать текст на абзацы. Отдельные слова можно выделять **жирным шрифтом** (используя с двух сторон две звездочки **) или **курсивом** (используя с двух сторон одну звездочку *)

Для начала нового абзаца необходимо начать писать с новой строки.

> Отдельно можно выделять примечания с помощью символа “>” в начале строки

Списки

Маркированные:

- Первый пункт
- Второй пункт
- Третий пункт

Так и нумерованные:

1. Первый пункт
2. Второй пункт
3. Третий пункт

Ссылки

Ссылки оформляются следующим образом:

[Яндекс](<https://ya.ru/>)

Картинки из интернета

Можно добавлять изображения как из интернета.

![Картинка из интернета] (<https://posit.co/wp-content/uploads/2022/10/rstudio-ide-dlpg-hero-test.png>)

Картинки с компьютера

Можно добавлять изображения с диска.

![Картинка с диска] ([templates/cheers.jpg](#))

Интерактивные компоненты

Основным преимуществом Rmarkdown является возможность встраивать код на языке R непосредственно в текст отчета. Далее рассмотрены возможности запуска интерактивных фрагментов кода на R в документе Rmarkdown.

Блоки кода R в документах Rmarkdown оформляются в блоках, называемых чанками.

```

```{r}
Здесь представлен код на R.
print("Hello")
```

```

При этом в зависимости от дополнительных параметров поведение кода в таких блоках может изменяться:

- *eval*: вывод в результирующий документ результатов вычисления блока кода;
- *echo*: вывод кода в результирующий документ;
- *warning*, *message*, и *error*: включить или отключить показ сообщений;
- *include*: включить блок кода в выходной документ;
- *fig.width* и *fig.height*: параметры вывода графиков в дюймах;
- *out.width* и *out.height*: размеры вывода элементов в документ;
- *fig.align*: выравнивание графика в документе;
- *fig.cap*: заголовок графика;
- *dev*: в каком виде выводить график;
- *child*: включение дочернего документа.

Каждый такой чанк можно запускать отдельно с помощью иконки с символом зеленой стрелки “Play”, чтобы посмотреть результат его работы. По умолчанию выводится как результат выполнения каждого чанка, так и код, определяющий его содержимое.

Результат работы чанка в RStudio представлен на **рисунке 8.119**.

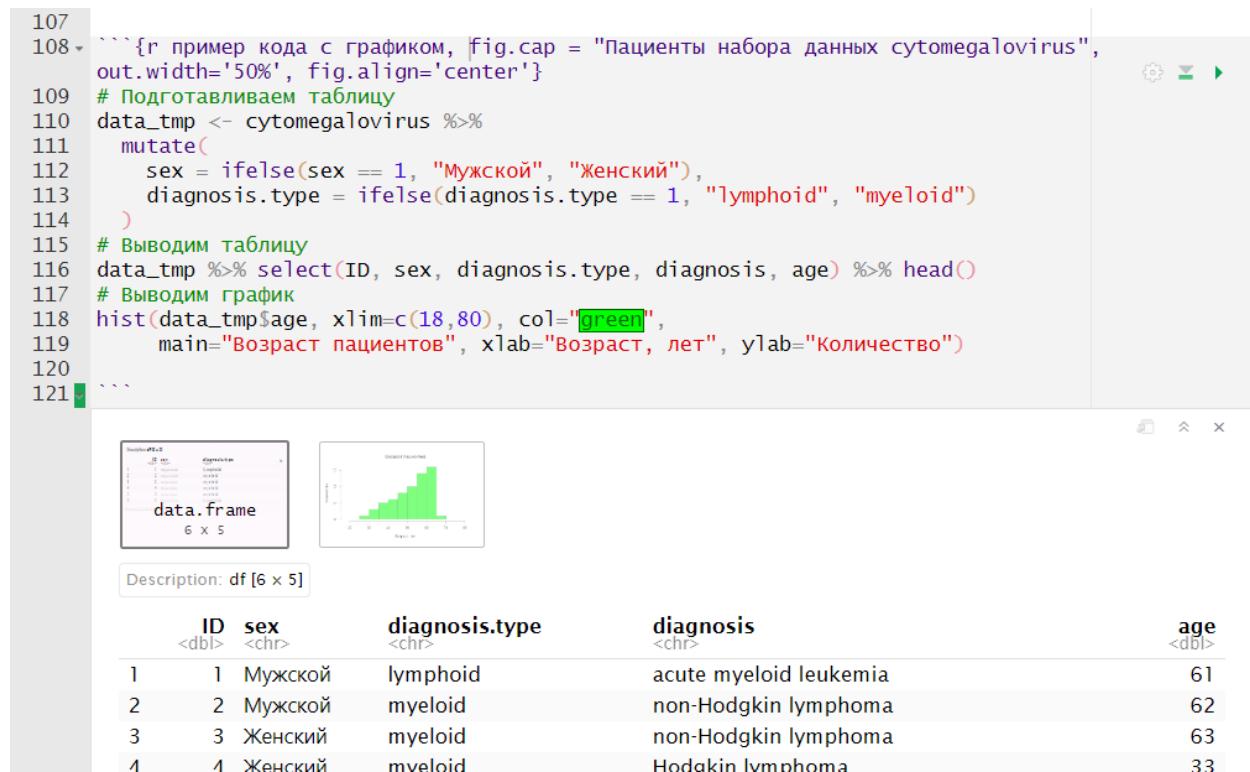


Рисунок 8.119. Результат работы чанка с кодом в программе Rstudio

Результат работы чанка в сгенерированном HTML-документе представлен на **рисунке 8.120**.

```
# Подготавливаем таблицу
data_tmp <- cytomegalovirus %>%
  mutate(
    sex = ifelse(sex == 1, "Мужской", "Женский"),
    diagnosis.type = ifelse(diagnosis.type == 1, "lymphoid", "myeloid")
  )
# Выводим таблицу
data_tmp %>% select(ID, sex, diagnosis.type, diagnosis, age) %>% head()
```

```
##   ID      sex diagnosis.type      diagnosis age
## 1  1 Мужской     lymphoid    acute myeloid leukemia 61
## 2  2 Мужской     myeloid    non-Hodgkin lymphoma 62
## 3  3 Женский     myeloid    non-Hodgkin lymphoma 63
## 4  4 Женский     myeloid    Hodgkin lymphoma 33
## 5  5 Женский     myeloid acute lymphoblastic leukemia 54
## 6  6 Мужской     lymphoid    myelofibrosis 55
```

```
# Выводим график
hist(data_tmp$age, xlim=c(18,80), col="green",
     main="Возраст пациентов", xlab="Возраст, лет", ylab="Количество")
```

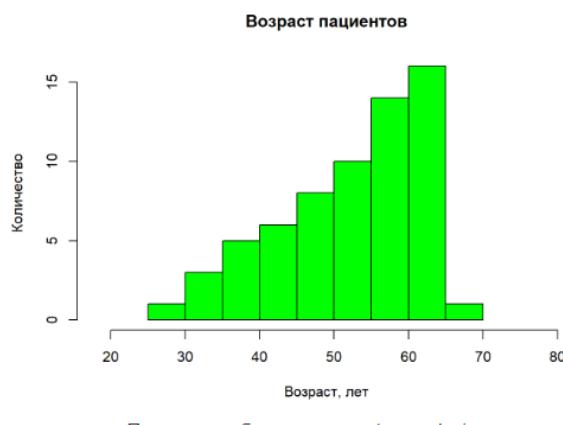


Рисунок 8.120. Результат работы чанка в HTML-документе

Для упрощения перехода между чанками можно задавать им псевдонимы. Например, на **рисунке 8.121**. представлены следующие псевдонимы: “пример кода с графиком”, “только код”, “только результат”.

The screenshot shows the RStudio interface with the 'Source' tab selected. A tooltip is displayed over line 108 of the code, listing various chunk options. The option 'Chunk 2: пример кода с графиком' is highlighted with a blue background.

```

## Пример кода на R
105
106 В примере ниже выводится и код, и результаты его выполнения
107
108 ``{r пример кода с графиком, fig.cap = "Пациенты набора", fig.width='50%', fig.align='center'}
109 # Подготавливаем таблицу
110 data_tmp <- cytomegalovirus %>%
111   заголовок второго уровня: текст
112   Заголовок третьего уровня: Списки
113   Заголовок третьего уровня: Ссылки
114   Картинки из интернета
115   Картинки с компьютера
116   Пример кода на R
117     Chunk 2: пример кода с графиком (selected)
118     Chunk 3: только код
119     Chunk 4: только результат
120   Вставка значения в текст
121   Chunk 5: значение в тексте
122   Использование скриптов из файлов для подготовки вычислений

```

Рисунок 8.121. Псевдонимы для чанков с кодом

Далее будут представлены основные режимы вывода результатов чанков с кодом (также см. [рисунок 8.122](#)).

Не включать в результирующий документ блок кода

```

```{r загрузка пакетов и функций, include=FALSE, message=FALSE}
Загрузка пакетов.
library(medicaldata)
library(tidyverse)
Создание функций
FtoC <- function(temp){
 c <- (temp - 32) * 5 / 9
 return(c)
}
Русификация reactable таблиц.
options(reactable.Language = reactableLang(
 pageSizeOptions = "показано {rows} значений",
 pageInfo = "с {rowStart} по {rowEnd} из {rows} строк",
 pagePrevious = "назад",
 pageNext = "вперед",
 searchPlaceholder = "Поиск...",
 noData = "Значения не найдены"
))
Настройки flextable по умолчанию.
set_flextable_defaults(
 font.family = "Arial", font.size = 10,
 border.color = "gray")
Вместо NA при выводе kable будем использовать пустую строку.

```

```

options(knitr.kable.NA = '')
Подготовка данных во внешнем файле.
source("./generate-data-table.R")
```

```

При установке атрибута *include=FALSE* данный блок кода не будет выводиться в результирующий документ. Это обычно используется для различных служебных блоков, где осуществляется загрузка пакетов, конфигурация функций, получение данных из внешних источников, либо запуск внешних файлов для вычислений - т.е. тех фрагментов кода, которые носят вспомогательный характер относительно генерируемого отчета.

Подразумевается, что документы Rmarkdown служат в основном для представления результатов, поэтому непосредственно вычисления стараются выносить во внешние файлы, а непосредственно в документе оставлять код, необходимый для корректного представления результатов.

Параметр *message=FALSE* показывает, что не нужно выводить информационные сообщения, которые выводит сессия R, например при загрузке пакетов.

Поведение чанка по умолчанию

По умолчанию, если в чанке отсутствуют дополнительные атрибуты, то выводится и код, и результаты его выполнения.

```

```{r код и результаты}
data_tmp %>% select(ID, sex, diagnosis.type, diagnosis, age) %>% glimpse()
```

data_tmp %>% select(ID, sex, diagnosis.type, diagnosis, age) %>% glimpse()

## #> #> #> #> #>
## #> Rows: 64
## #> Columns: 5
## #> $ ID <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, ...
## #> $ sex <chr> "Мужской", "Мужской", "Женский", "Женский", ...
## #> $ diagnosis.type <chr> "lymphoid", "myeloid", "myeloid", "myeloid", ...
## #> $ diagnosis <chr> "acute myeloid leukemia", "non-Hodgkin lymphoma", "non-...
## #> $ age <dbl> 61, 62, 63, 33, 54, 55, 67, 51, 44, 59, 45, 57, 52, 38, ...

```

Рисунок 8.122. Результат выполнения чанка с кодом

Вывод только кода, без результата выполнения

```

```{r только код, eval = FALSE}
data_tmp %>% select(ID, sex, diagnosis.type, diagnosis, age) %>% glimpse()
```

data_tmp %>% select(ID, sex, diagnosis.type, diagnosis, age) %>% glimpse()

```

Рисунок 8.123. Вывод чанка с кодом, без результата его исполнения

Вывод только результата выполнения кода

```

```{r только результатом, echo = FALSE}
data_tmp %>% select(ID, sex, diagnosis.type, diagnosis, age) %>% glimpse()
```

## Rows: 64
## Columns: 5
## $ ID      <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, ...
## $ sex     <chr> "Мужской", "Мужской", "Женский", "Женский", ...
## $ diagnosis.type <chr> "lymphoid", "myeloid", "myeloid", "myeloid", ...
## $ diagnosis <chr> "acute myeloid leukemia", "non-Hodgkin lymphoma", "non-...
## $ age      <dbl> 61, 62, 63, 33, 54, 55, 67, 51, 44, 59, 45, 57, 52, 38, ...

```

Рисунок 8.124. Результат выполнения чанка с кодом, без вывода самого кода

Вставка значений в текст

Результат вычислений можно сохранить в переменную, чтобы потом иметь возможность вставить ее значение в любое место в тексте документа Rmarkdown. В примере, представленном ниже, происходит расчет количества строк в таблице *data_tmp* (где одна строка = один пациент) с записью результата в переменную *patients_count*. Далее переменную *patients_count* можно вставить в любое место документа вместе с основным текстом с помощью команды *r patients_count*. Такой подход позволяет автоматически вставлять результаты вычислений в текстовые предложения и таким образом шаблонизировать отчет. Например, предложение “В исследовании участвовало `r patients_count` пациентов” при генерации документа автоматически превратится в предложение, отражающее число пациентов (**рисунок 8.125.**).

```
```{r значение в тексте}
```

```
patients_count <- nrow(data_tmp)
```

*В исследовании участвовало `r patients\_count` пациентов.*

Результат такой вставки можно увидеть в сгенерированном документе.

```
patients_count <- nrow(data_tmp)
```

*В исследовании участвовало 64 пациентов.*

**Рисунок 8.125.** Автоподстановка значений в генерируемый текст

Таблицы

Таблицы можно выводить так же, как они обычно выводятся в консоль в Rstudio.

```
```{r вывод таблицы, echo=FALSE}
data
```
```

```

A tibble: 7 × 5
diagnosis sex n percent_diag percent_overall
<chr> <chr> <int> <dbl> <dbl>
1 lymphoid Женский 15 50 23.4
2 lymphoid Мужской 15 50 23.4
3 myeloid Женский 13 46.4 20.3
4 myeloid Мужской 15 53.6 23.4
5 <NA> Женский 2 33.3 3.12
6 <NA> Мужской 4 66.7 6.25
7 Всего Всего 64 NA 100

```

**Рисунок 8.126.** Вывод таблиц в документе Rmarkdown в “консольном” формате

Для отчета такое представление таблиц является не совсем привычным, однако существует еще несколько вариантов вывода таблиц.

#### Таблицы kable

По умолчанию в RMarkdown для вывода таблиц используется функция *kable()* из пакета *knitr*. Функция принимает большое количество аргументов, позволяющих определить внешний вид таблицы. Далее в примере будут использованы некоторые из аргументов: название таблицы, заголовки столбцов и настройки выравнивания.

```

```{r table-kable, echo=FALSE, out.width='100%'}

data %>%
  knitr::kable(caption = 'Распределение пациентов по диагнозам',
               col.names = c("Диагноз", "Пол", "Случаев", "В группе", "Всего"),
               align = "llcrr", digits = 2)
```

```

Здесь в качестве атрибутов функции *knitr::kable()* используются:

- *caption* - заголовок таблицы;
- *col.names* - заголовки столбцов таблицы;
- *align* - выравнивание столбцов таблицы, где каждая буква обозначает столбец. В данном примере происходит выравнивание первых два столбца по левому краю (два символа “l”), средний - по центру (символ “c”), два последних - по правому (символ “r”);
- *digits* - определяет количество символов после запятой в дробных числах.

Результат генерации документа представлен на **рисунке 8.127..**

## Распределение пациентов по диагнозам

| Диагноз  | Пол     | Случаев | В группе | Всего  |
|----------|---------|---------|----------|--------|
| lymphoid | Женский | 15      | 50.00    | 23.44  |
| lymphoid | Мужской | 15      | 50.00    | 23.44  |
| myeloid  | Женский | 13      | 46.43    | 20.31  |
| myeloid  | Мужской | 15      | 53.57    | 23.44  |
|          | Женский | 2       | 33.33    | 3.12   |
|          | Мужской | 4       | 66.67    | 6.25   |
| Всего    | Всего   | 64      |          | 100.00 |

**Рисунок 8.127.** Генерация таблицы в Rmarkdown с помощью функции *knitr()*

Следует обратить внимание, что в результирующей таблице отсутствуют значения NA, потому что в чанке инициализации была указана соответствующая настройка.

```
options(knitr.kable.NA = '')
```

## Таблицы kableExtra

Расширенные возможности по настройке стиля таблиц для формата RMarkdown содержит пакет *kableExtra*. Кроме того, в данном пакете содержатся дополнительные предустановленные стили для оформления kable-таблиц. Далее рассмотрены некоторые стили.

### Стиль таблиц с подсветкой строк при наведении

Данный стиль имеет смысл использовать для интерактивных отчетов, например HTML-документов (**рисунок 8.128.**).

```
```{r kable-hover, echo=FALSE}
data %>%
  kbl(caption = 'Распределение пациентов по диагнозам',
       col.names = c("Диагноз", "Пол", "Случаев", "В группе", "Всего"),
       align = "llcrr", digits = 2) %>%
  kable_paper("hover", full_width = F)
```
```

Распределение пациентов по диагнозам

| Диагноз  | Пол     | Случаев | В группе | Всего  |
|----------|---------|---------|----------|--------|
| lymphoid | Женский | 15      | 50.00    | 23.44  |
| lymphoid | Мужской | 15      | 50.00    | 23.44  |
| myeloid  | Женский | 13      | 46.43    | 20.31  |
| myeloid  | Мужской | 15      | 53.57    | 23.44  |
|          | Женский | 2       | 33.33    | 3.12   |
|          | Мужской | 4       | 66.67    | 6.25   |
| Всего    | Всего   | 64      |          | 100.00 |

**Рисунок 8.128.** Генерация таблицы в Rmarkdown с подсветкой строк с помощью функции *kable\_paper()* из пакета *kableExtra*

*Стиль таблиц для печати*

Данные стили часто применяются в научных статьях и книгах (**рисунки 8.129.-8.130.**).

```
```{r kable-classic, echo=FALSE, out.width='100%'}
data %>%
  kbl(caption = 'Распределение пациентов по диагнозам',
       col.names = c("Диагноз", "Пол", "Случаев", "В группе", "Всего"),
       align = "llcrr", digits = 2) %>%
  kable_classic(full_width = F, html_font = "Cambria")
```

```

Распределение пациентов по диагнозам

| Диагноз  | Пол     | Случаев | В группе | Всего  |
|----------|---------|---------|----------|--------|
| lymphoid | Женский | 15      | 50.00    | 23.44  |
| lymphoid | Мужской | 15      | 50.00    | 23.44  |
| myeloid  | Женский | 13      | 46.43    | 20.31  |
| myeloid  | Мужской | 15      | 53.57    | 23.44  |
|          | Женский | 2       | 33.33    | 3.12   |
|          | Мужской | 4       | 66.67    | 6.25   |
| Всего    | Всего   | 64      |          | 100.00 |

**Рисунок 8.129.** Генерация таблицы в Rmarkdown для печати с помощью функции *kable\_classic()* из пакета *kableExtra*

```

```{r kable-classic2, echo=FALSE, out.width='100%'}
data %>%
  kbl(caption = 'Распределение пациентов по диагнозам',
       col.names = c("Диагноз", "Пол", "Случаев", "В группе", "Всего"),
       align = "llcrr", digits = 2) %>%
  kable_classic_2(full_width = F)
```

```

Распределение пациентов по диагнозам

| Диагноз  | Пол     | Случаев | В группе | Всего  |
|----------|---------|---------|----------|--------|
| lymphoid | Женский | 15      | 50.00    | 23.44  |
| lymphoid | Мужской | 15      | 50.00    | 23.44  |
| myeloid  | Женский | 13      | 46.43    | 20.31  |
| myeloid  | Мужской | 15      | 53.57    | 23.44  |
|          | Женский | 2       | 33.33    | 3.12   |
|          | Мужской | 4       | 66.67    | 6.25   |
| Всего    | Всего   | 64      |          | 100.00 |

**Рисунок 8.130.** Генерация таблицы в Rmarkdown для печати с помощью функции *kable\_classic2()* из пакета *kableExtra*

### Таблицы flextable

Таблицы, созданные с помощью пакета *kable*, хорошо отображаются в HTML-документах, но могут вызывать проблемы при вставке в документы Word, генерируемые из RMarkdown. Для создания таблиц, которые будут корректно отображаться в Word (и PowerPoint), рекомендуется использовать пакет *flextable*. Он предлагает более универсальный подход, гарантирующий одинаковый внешний вид таблицы вне зависимости от способа рендеринга (**рисунок 8.131.**).

```

```{r flextable, echo=FALSE}
ft_table
```

```

## Типы диагнозов

| Вторая строка заголовка |         |         | Процент               |                    |
|-------------------------|---------|---------|-----------------------|--------------------|
| Диагноз                 | Пол     | Случаев | в группе <sup>1</sup> | всего <sup>2</sup> |
|                         |         |         |                       |                    |
| <i>Lymphoid</i>         | Женский | 15      | 50.00%                | 23.44%             |
|                         | Мужской | 15      | 50.00%                | 23.44%             |
| <i>Myeloid</i>          | Женский | 13      | 46.43%                | 20.31%             |
|                         | Мужской | 15      | 53.57%                | 23.44%             |
|                         | Женский | 2       | 33.33%                | 3.12%              |
|                         | Мужской | 4       | 66.67%                | 6.25%              |
| <b>Всего</b>            |         |         | <b>100.00%</b>        |                    |

*На основании данных из набора сутомегаловирус*

<sup>1</sup>Доля пациента в рамках диагноза

<sup>2</sup>Доля пациентов в рамках исследования

**Рисунок 8.131.** Генерация таблицы в Rmarkdown с помощью пакета *flextable*

Таблицы gt

Таблицы gt изначально разрабатывались с учетом отображения в HTML и для работы с Rmarkdown, поэтому их использование максимально оправдано.

```
```{r gt, echo=FALSE}
gt_table
```
```

## Типы диагнозов

Вторая строка заголовка

| Диагноз             | Пол     | Случаев   | Процент               |                    |
|---------------------|---------|-----------|-----------------------|--------------------|
|                     |         |           | в группе <sup>1</sup> | всего <sup>2</sup> |
| <i>Lymphoid</i>     | Женский | 15        | 50.00%                | 23.44%             |
|                     | Мужской | 15        | 50.00%                | 23.44%             |
| <i>Myeloid</i>      | Женский | 13        | 46.43%                | 20.31%             |
|                     | Мужской | 15        | <b>53.57%</b>         | 23.44%             |
| <i>Неуточненный</i> | Женский | 2         | 33.33%                | 3.12%              |
|                     | Мужской | 4         | <b>66.67%</b>         | 6.25%              |
| <b>Всего</b>        |         | <b>64</b> |                       | <b>100.00%</b>     |

<sup>1</sup> Доля пациента в рамках диагноза

<sup>2</sup> Доля пациентов в рамках исследования

На основании данных из набора `cytomegalovirus`

Рисунок 8.132. Генерация таблицы в Rmarkdown с помощью пакета *gt*

### Интерактивные таблицы

Если в результате генерации документа Rmarkdown планируется получить HTML-документ, то можно использовать интерактивные компоненты. Это может быть полезно, когда необходимо вывести достаточно большую таблицу, но отображать ее сразу целиком не имеет смысла. В этом случае можно вывести только первые 10 строк, а просмотр остальных организовать с помощью сортировки, фильтров и пагинации (переключения по страницам). Для реализации такой механики можно использовать пакет *reactable*.

Перед использованием интерактивных таблиц рекомендуется указать настройки перевода элементов управления на русский язык. Эта настройка делается один раз для всего документа.

```
options(reactable.Language = reactableLang(
 pageSizeOptions = "показано {rows} значений",
 pageInfo = "с {rowStart} по {rowEnd} из {rows} строк",
```

```

pagePrevious = "назад",
pageNext = "вперед",
searchPlaceholder = "Поиск...",
noData = "Значения не найдены"
))

```

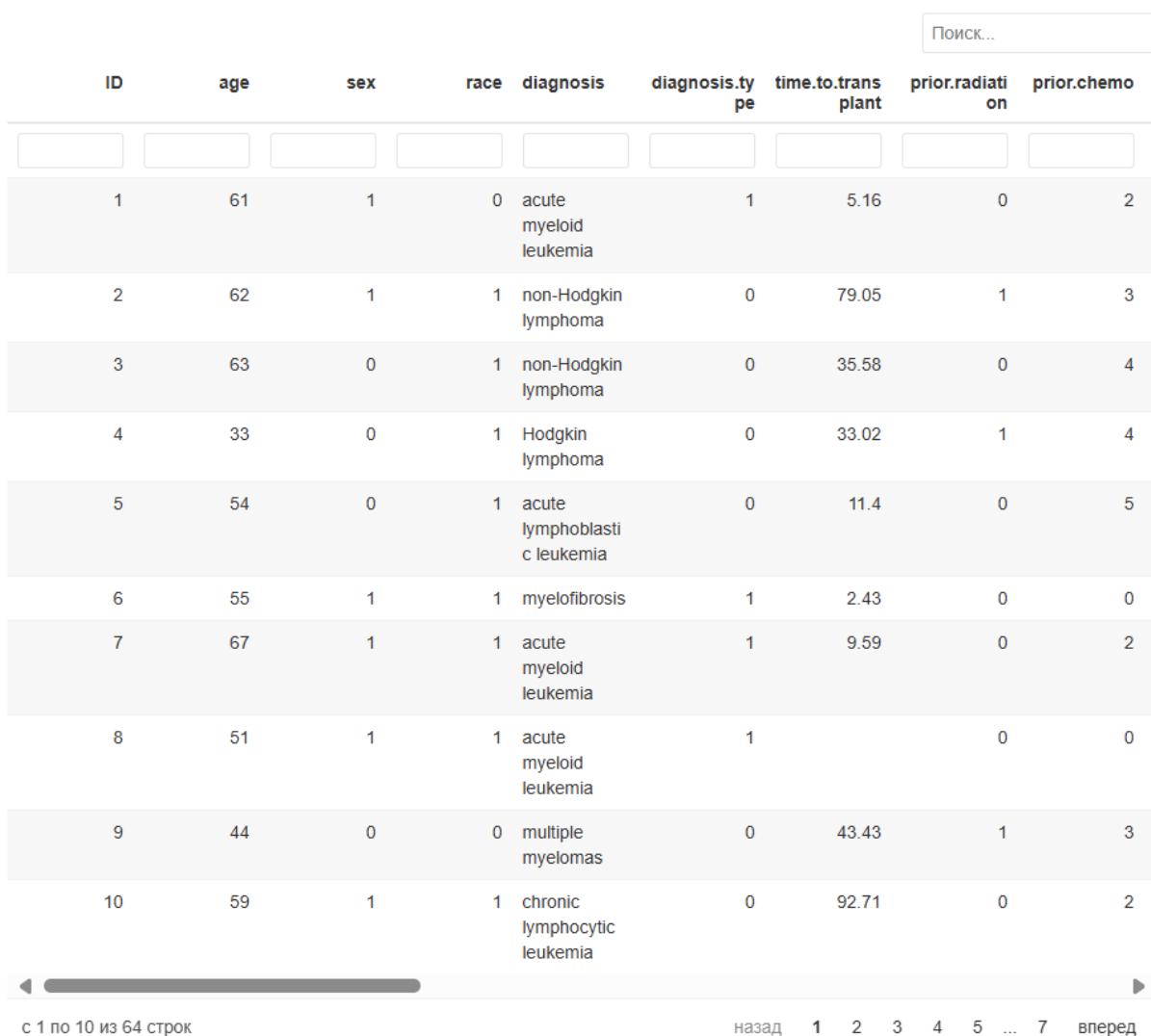
При создании интерактивной таблицы можно установить множество различных дополнительных параметров, однако в рамках примера будет использован ограниченный их перечень: включение фильтров, сортировки и поиска, а также добавление чересстрочной заливки строк для лучшего восприятия. Далее представлен код для отображения набора данных *cytomegalovirus* (набор данных рассматривался ранее в [разделе 2.6.1.](#)) в виде такой таблицы.

```

```{r reactable, echo=FALSE, out.width='100%'}
reactable(cytomegalovirus, filterable = TRUE, searchable = TRUE, striped = TRUE)
```

```

В результате будет получена интерактивная таблица, в которой можно переходить по страницам и искать интересующие данные.



The image shows an interactive R Shiny table for the 'cytomegalovirus' dataset. The table has 11 columns: ID, age, sex, race, diagnosis, diagnosis.type, time.to.transplant, prior.radiation, and prior.chemotherapy. The first row contains empty input fields for filtering. Rows 1 through 10 show specific patient data. Row 1: ID 1, age 61, sex 1, race 0, diagnosis acute myeloid leukemia, diagnosis.type 1, time.to.transplant 5.16, prior.radiation 0, prior.chemotherapy 2. Row 2: ID 2, age 62, sex 1, race 1, diagnosis non-Hodgkin lymphoma, diagnosis.type 0, time.to.transplant 79.05, prior.radiation 1, prior.chemotherapy 3. Row 3: ID 3, age 63, sex 0, race 1, diagnosis non-Hodgkin lymphoma, diagnosis.type 0, time.to.transplant 35.58, prior.radiation 0, prior.chemotherapy 4. Row 4: ID 4, age 33, sex 0, race 1, diagnosis Hodgkin lymphoma, diagnosis.type 0, time.to.transplant 33.02, prior.radiation 1, prior.chemotherapy 4. Row 5: ID 5, age 54, sex 0, race 1, diagnosis acute lymphoblastic leukemia, diagnosis.type 0, time.to.transplant 11.4, prior.radiation 0, prior.chemotherapy 5. Row 6: ID 6, age 55, sex 1, race 1, diagnosis myelofibrosis, diagnosis.type 1, time.to.transplant 2.43, prior.radiation 0, prior.chemotherapy 0. Row 7: ID 7, age 67, sex 1, race 1, diagnosis acute myeloid leukemia, diagnosis.type 1, time.to.transplant 9.59, prior.radiation 0, prior.chemotherapy 2. Row 8: ID 8, age 51, sex 1, race 1, diagnosis acute myeloid leukemia, diagnosis.type 1, time.to.transplant 43.43, prior.radiation 0, prior.chemotherapy 0. Row 9: ID 9, age 44, sex 0, race 0, diagnosis multiple myelomas, diagnosis.type 0, time.to.transplant 92.71, prior.radiation 1, prior.chemotherapy 3. Row 10: ID 10, age 59, sex 1, race 1, diagnosis chronic lymphocytic leukemia, diagnosis.type 0, time.to.transplant 0, prior.radiation 0, prior.chemotherapy 2. A search bar at the top right says 'Поиск...'. At the bottom, there are navigation controls: a left arrow, a long horizontal scrollbar, a right arrow, page numbers 1, 2, 3, 4, 5, ..., 7, ..., and a right arrow, and a message 'с 1 по 10 из 64 строк'.

| ID | age | sex | race | diagnosis                    | diagnosis.type | time.to.transplant | prior.radiation | prior.chemotherapy |
|----|-----|-----|------|------------------------------|----------------|--------------------|-----------------|--------------------|
|    |     |     |      |                              |                |                    |                 |                    |
| 1  | 61  | 1   | 0    | acute myeloid leukemia       | 1              | 5.16               | 0               | 2                  |
| 2  | 62  | 1   | 1    | non-Hodgkin lymphoma         | 0              | 79.05              | 1               | 3                  |
| 3  | 63  | 0   | 1    | non-Hodgkin lymphoma         | 0              | 35.58              | 0               | 4                  |
| 4  | 33  | 0   | 1    | Hodgkin lymphoma             | 0              | 33.02              | 1               | 4                  |
| 5  | 54  | 0   | 1    | acute lymphoblastic leukemia | 0              | 11.4               | 0               | 5                  |
| 6  | 55  | 1   | 1    | myelofibrosis                | 1              | 2.43               | 0               | 0                  |
| 7  | 67  | 1   | 1    | acute myeloid leukemia       | 1              | 9.59               | 0               | 2                  |
| 8  | 51  | 1   | 1    | acute myeloid leukemia       | 1              |                    | 0               | 0                  |
| 9  | 44  | 0   | 0    | multiple myelomas            | 0              | 43.43              | 1               | 3                  |
| 10 | 59  | 1   | 1    | chronic lymphocytic leukemia | 0              | 92.71              | 0               | 2                  |

**Рисунок 8.133.** Генерация интерактивной таблицы в Rmarkdown с помощью пакета *reactable*

## Интерактивные карты

С помощью пакета *leaflet* в отчет можно добавлять интерактивные карты. В примере кода ниже представлено отображение карты со следующими параметрами:

- отображение во всю ширину страницы;
- указание начальной позиции на карте и уровня приближения (с помощью функции *setView()*);
- добавление маркера с подписью (с помощью функции *addPopups()*).

```
```{r out.width='100%', echo=FALSE}
Leaflet() %>% addTiles() %>%
  setView(32.06006102596806, 54.76838259349348, zoom = 17) %>%
  addPopups(
    32.06006102596806, 54.76838259349348,
    'Смоленский государственный медицинский университет'
  )
...``
```

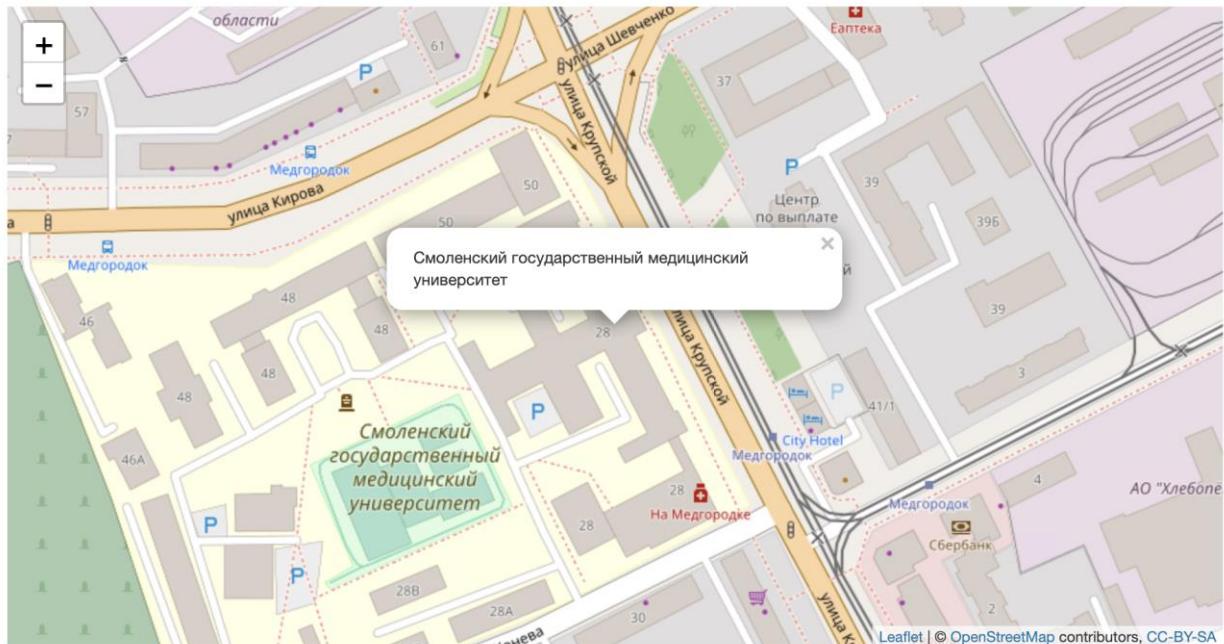
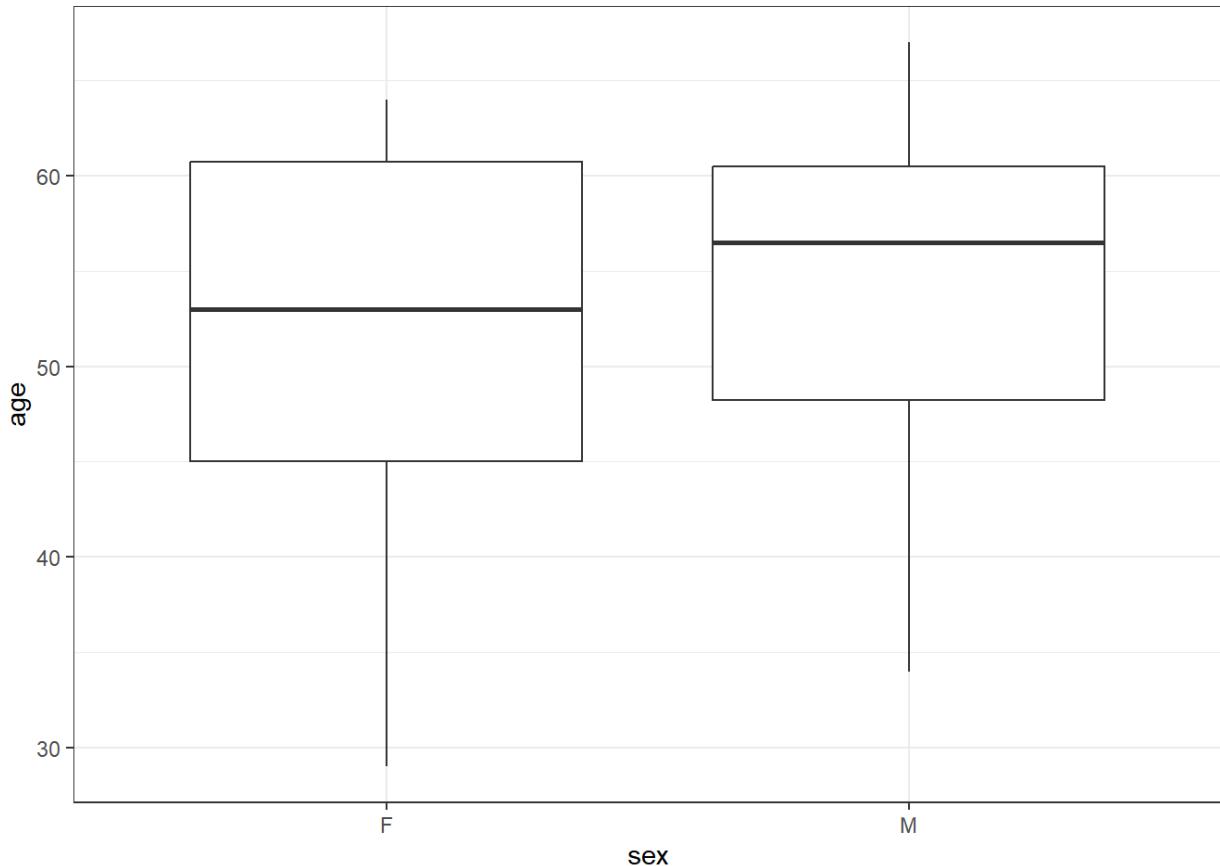


Рисунок 8.134. Генерация интерактивной карты в Rmarkdown с помощью пакета *leaflet*

Графики

С помощью пакетов *ggplot2* и *plotly* можно добавлять в отчеты интерактивные графики, которые предлагают более глубокое взаимодействие с данными. *ggplot2* предоставляет широкие возможности для создания статических графиков, которые можно преобразовать в интерактивные с помощью *plotly*. Обычные графики *ggplot2* отображаются в отчете как изображения, в то время как графики *plotly* остаются интерактивными в HTML-документах.

```
```{r ggplot2, echo=FALSE}
gg <- ggplot(cytomegalovirus %>% mutate(sex = ifelse(sex == 1, "M", "F")),
aes(sex, age)) + geom_boxplot() + theme_bw()
gg
```



**Рисунок 8.135.** Генерация графика в Rmarkdown с помощью пакета *ggplot2*

Если же “обернуть” созданный график в функцию *ggplotly()* из пакета *plotly*, то он станет интерактивным: его можно увеличивать, смотреть значения, переключать интересующие ряды и так далее.

```
```{r plotly, echo=FALSE}
ggplotly(gg)
```

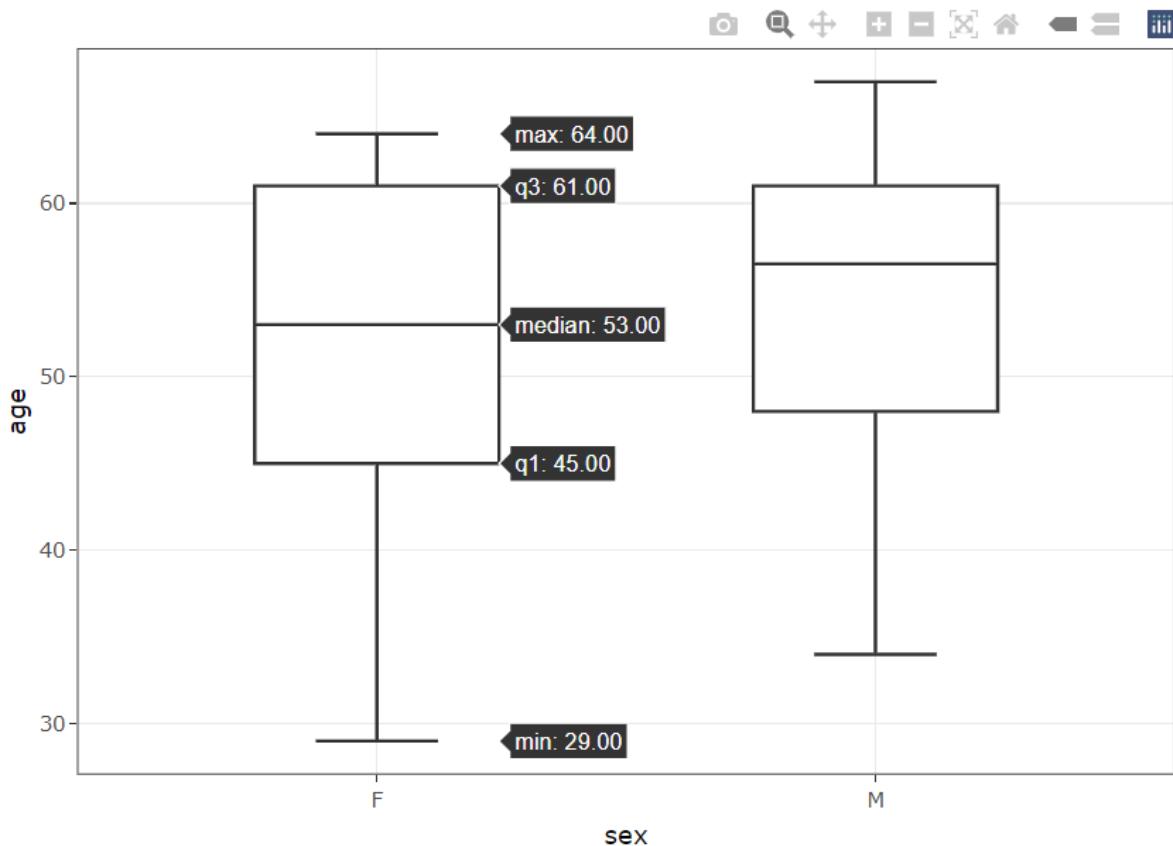


Рисунок 8.136. Генерация графика в Rmarkdown с помощью функции *ggplotly()* на основе графика *ggplot2*

Генерация отчета

После создания отчета в формате Rmarkdown его нужно сгенерировать в какой-либо целевой формат. Для этого служит кнопка Knit на панели управления RStudio (**рисунок 8.137.**).

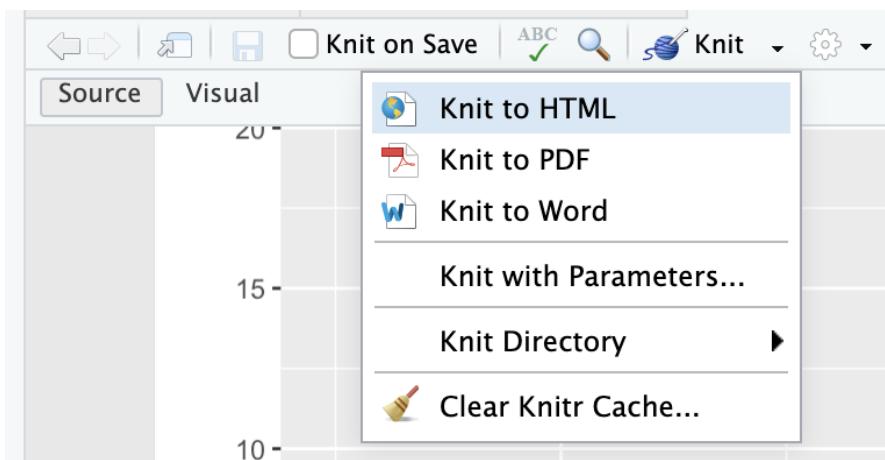


Рисунок 8.137. Интерфейс программы RStudio с меню для генерации различных документов на основе Rmarkdown

Количество пунктов может варьироваться в зависимости от преамбулы документа. При желании можно отметить галочкой пункт “Knit on Save”, чтобы отчеты генерировались при каждом сохранении Rmarkdown документа. Стоит отметить, что все интерактивные элементы оптимальным образом генерируются только в формате HTML. Сгенерированные файлы сохраняются в папке проекта. При генерации в формат Word рекомендуется удалить все интерактивные карты, графики, а для вывода таблиц использовать пакет *flextable*. В этом случае после генерации можно получить следующий документ.

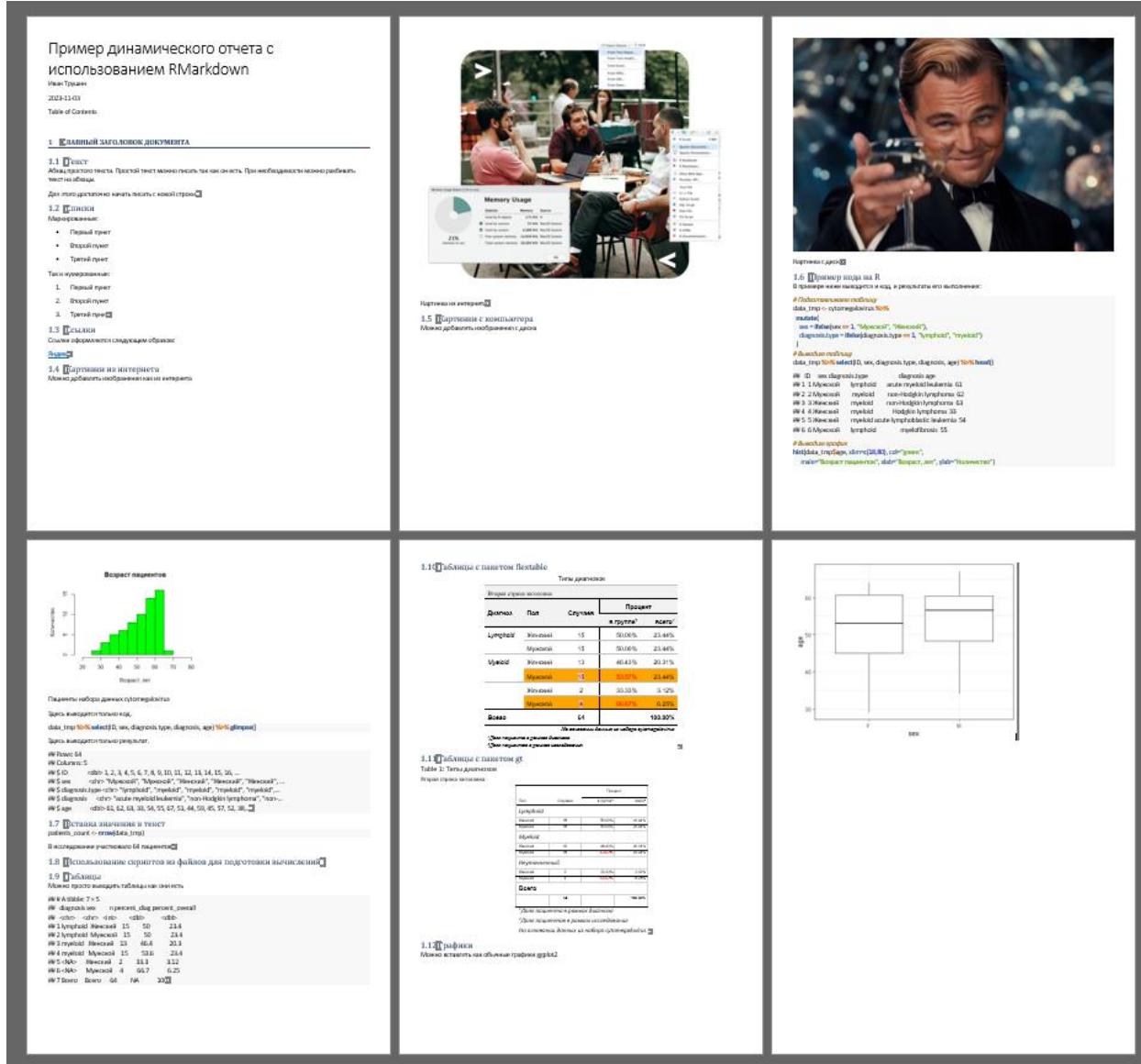


Рисунок 8.138. Результат генерации документа в формате Word на основе Rmarkdown

Параметризованные отчеты

Довольно часто возникает ситуация, когда необходимо создать несколько однотипных отчетов, но для разных срезов данных - например по разным центрам одного

исследования, различным группам пациентов, за различные временные периоды. При этом непосредственно форма и содержание отчета должно оставаться неизменным.

В этом случае могут помочь параметризованные отчеты Rmarkdown. В преамбуле отчета объявляется дополнительный блок *params*, в котором можно определить несколько параметров, которые будут использоваться в процессе формирования отчета. Например, зададим параметр *sex*, который будет определять, будет ли отчет содержать данные по всем пациентам или по конкретному полу.

```
---
```

```
title: "Пример отчета с использованием RMarkdown"
author: "Иван Трушин"
date: "2022-01-03"
params:
  sex: ""
output:
  html_document:
    toc: yes
    df_print: kable
always_allow_html: yes
---
```

По умолчанию для параметра *sex* будет использоваться значение пустой строки. Теперь непосредственно в самом отчете нужно добавить логику, определяющую поведение нашего отчета, в зависимости от входного параметра. Сделаем это на этапе фильтрации основного набора данных. Если будет передана пустая строка, мы используем все записи из набора данных, если нет — фильтруем по значению соответствующего поля.

```
```{r пример кода с графиком, echo = FALSE, eval = FALSE}
Фильтруем таблицу в зависимости от параметров
if (params$sex != "") {
 data_tmp <- data_tmp %>% filter(sex == params$sex)
}
````
```

Теперь можно запустить генерацию отчета с помощью команды.

```
rmarkdown::render(
  input = "2.6.Report.Static.Rmd",
  output_file = "2.6.Report.Static.html",
  params = List(sex = "Мужской"),
  envir = parent.frame()
)
```

Преимуществом данного подхода является возможность создания сразу нескольких отдельных отчетов для разного набора параметров таким образом, чтобы результат выполнения каждого отчета был сохранен в отдельном файле.

Для этого необходимо заранее определить список наборов параметров для формирования отчетов.

```
params_list <- list(  
  list(sex = ""),  
  list(sex = "Мужской"),  
  list(sex = "Женский")  
)
```

Затем в цикле запустить выполнение отчета для каждого из набора параметров, динамически определяя имя сгенерированного файла.

```
for (par in params_list) {  
  rmarkdown::render(  
    input = "2.6.Report.Static.Rmd",  
    output_file = paste0("2.6.Report.Static_", par$sex, ".html"),  
    output_format = "html_document",  
    output_dir = ".",  
    params = par,  
    envir = parent.frame()  
)  
}
```

В результате у получится три файла отчета, каждый из которых будет содержать данные только по своему срезу набора данных.

```
2.6.Report.Static.Rmd  
2.6.Report.Static_.html  
2.6.Report.Static_Женский.html  
2.6.Report.Static_Мужской.html
```

Публикация отчета

Отчеты Rmarkdown (в том числе с интерактивными элементами) можно бесплатно публиковать на сайте <https://rpubs.com>, чтобы поделиться им с другими пользователями и всеми, у кого есть ссылка. Для публикации отчета в панели инструментов RStudio есть отдельная кнопка (**рисунок 8.139.**).

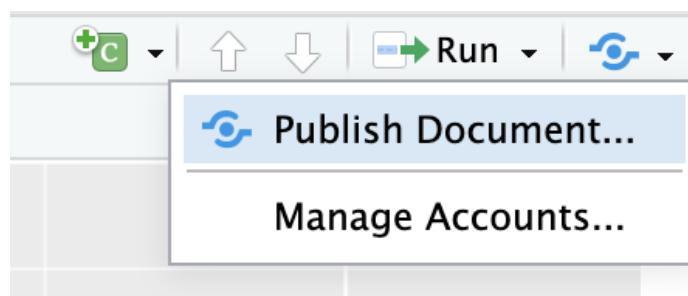


Рисунок 8.139. Кнопка для публикации документа Rmarkdown на сайте <https://rpubs.com>

Чтобы публиковать отчеты на сайте <https://rpubs.com>, необходимо создать учетную запись. После регистрации все опубликованные отчеты будут храниться в указанном профиле на сайте.

8.2.8. Фабрика отчетов

Довольно часто, особенно в сфере исследований, отчеты должны генерироваться регулярно с заданной периодичностью (раз в неделю, месяц и т.д.). При этом сам формат их не меняется, меняется только набор данных.

Для оптимизации и структуризации этой задачи существует пакет *reportfactory*, который предлагает заранее определенную структуру проекта для хранения данных, скриптов их обработки, шаблонов отчетов и механизм их генерации. Данный пакет позволяет создать "фабрику отчетов", которая по команде запускает генерацию отчетов в формате Rmarkdown, автоматически создает папки для выходных данных с датой и временем в названии, что позволяет легко контролировать версии.

При этом сохраняется исходный шаблон отчета, поэтому всегда можно восстановить как именно были получены результаты на определенный момент времени, даже в случае если основной шаблон был изменен.

После установки и загрузки пакета *reportfactory* создание новой фабрики можно выполнить с помощью функции *new_factory()* в консоли R.

```
install.packages("reportfactory")
library(reportfactory)
new_factory()
```

После этого будет создана новая папка *new_factory* с проектом *new_factory.Rproj* и созданным git-репозиторием. RStudio автоматически переключится на работу с ним.

При желании с помощью дополнительных параметров можно изменить имя папки и проекта при создании.

```
reportfactory::new_factory(
  # имя для папки фабрики
  factory = "MyFactory",
  # путь к файлу для новой фабрики
  path = "./my_factory",
  # альтернативное имя для папки, в которой будут содержаться отчеты Rmarkdown
  report_sources = "report_templates",
  # альтернативное имя для папки, в которой содержатся выходные результаты отчетов
  outputs = "report_output"
)
```

Можно сразу создать файл *RUN.R*, в котором загрузить пакет *library(reportfactory)*, чтобы запускать команды работы с фабрикой не только через консоль, но и сохранить их для дальнейшего использования.

По умолчанию структура созданных файлов и папок выглядит следующим образом (ее можно посмотреть с помощью функции *factory_overview()*):

```
factory_overview()
/home/code/new_factory
├── .gitignore
├── .here
├── README.md
└── RUN.R
├── data
│   ├── clean
│   └── raw
│       └── example_data.csv
├── factory_config
├── new_factory.Rproj
├── outputs
├── report_sources
│   └── example_report.Rmd
└── scripts
```

- Файлы *.gitignore* и *README.md* для связи проекта с репозиторием git.
- Файл *RUN.R* - здесь описываются команды для работы с фабрикой.
- Файл *.here* - нужен для работы пакета *here*, чтобы было удобнее ссылаться на все файлы в рамках проекта с помощью составных путей к файлам относительно папки проекта.
- Папка *data* - может быть использована для хранения данных. Она включает в себя папки *raw* для хранения "сырых" и *clean* для хранения очищенных данных. Предполагается, что в папку *raw* попадают файлы из внешних систем без изменений, как есть. Затем они проходят подготовку с помощью скриптов из папки *scripts* и попадают в папку *clean*.
- Файл *factory_config* - содержит настройки проекта для фабрики, определяет назначение папок.
- Папка *outputs* - здесь будут появляться сгенерированные отчеты (HTML, Word, PDF).
- Папка *report_sources* - папка для хранения файлов Rmarkdown, которые будут формировать отчеты. Для примера она уже содержит файл *example_report.Rmd*;
- Папка *scripts* - предназначена для хранения скриптов R, которые обрабатывают наборы данных, на основе которых потом будут формироваться отчеты. Здесь можно разместить логику получения данных, очистки, валидации, трансформации - все то, что не влияет на вывод результата непосредственно. Также здесь могут быть сохранены скрипты, которые вызываются из отчетов.

Проект построен таким образом, чтобы в рамках репозитория можно было отслеживать изменения как скриптов, так и данных. Кроме того, при такой структуре достаточно просто организовать процесс генерации отчетов, в котором достаточно положить в папку `raw` новые данные и запустить генерацию отчетов всего одной командой.

Для примера далее будет подготовлена фабрика для создания отчетов по набору данных `cytomegalovirus` (набор данных рассматривался ранее в разделе 2.6.1.).

В папку `data/raw` следует разместить сохраненный набор данных в формате `csv` — как будто это сырье данные, полученные от системы сбора данных. Затем подготовить скрипт `prepare-clean-data.R`, который будет содержать команды для обработки данных. В нем можно заметить пример использования пакета `here`, который формирует путь к файлу для чтения. Благодаря этому пакету неважно, где относительно проекта находится файл скрипта или файл данных — все пути можно определять относительно папки проекта.

```
# Загрузка нужных пакетов.
library(dplyr)
library(here)
# Прочтение сырых данных.
raw <- read.csv2(here("data", "raw", "cytomegalovirus.csv"))
```

Перекодировка данных сырого набора согласно официальному словарю данного набора (<https://www.causeweb.org/tshs/datasets/Cytomegalovirus Data Dictionary.pdf>).

```
data <- raw %>%
  mutate(
    sex = recode(sex, `1` = "Male", `0` = "Female"),
    race = recode(race, `1` = "White", `0` = "African American"),
    diagnosis.type = recode(diagnosis.type, `1` = "Myeloid", `0` = "Lymphoid"),
    prior.radiation = recode(prior.radiation, `0` = "No", `1` = "Yes"),
    prior.transplant = recode(prior.transplant, `0` = "No", `1` = "Yes"),
    recipient.cmv = recode(recipient.cmv, `0` = "Negative", `1` = "Positive"),
    donor.cmv = recode(donor.cmv, `0` = "Negative", `1` = "Positive"),
    donor.sex = recode(donor.sex, `1` = "Male", `0` = "Female"),
    C1.C2 = recode(C1.C2, `1` = "Homozygous", `0` = "Heterozygous"),
    cmv = recode(cmv, `0` = "No", `1` = "Yes"),
    agvh = recode(agvh, `0` = "No", `1` = "Yes"),
    cgvh = recode(cgvh, `0` = "No", `1` = "Yes")
  )
```

Дополнительно будут определены группы возраста.

```
data <- data %>%
  mutate(
    age.group = case_when(
      age < 18 ~ "детский",
      age >= 18 & age <= 44 ~ "молодой",
      age >= 45 & age <= 59 ~ "средний",
```

```

age >= 60 & age <= 74 ~ "пожилой",
age >= 75 & age <= 90 ~ "старческий",
age > 90 ~ "долголетие",
is.na(age) ~ "нет данных"
)
) %>%
relocate(age.group, .after = age)

```

На данном этапе можно условно считать, что все процедуры очистки, подготовки и валидации данных описаны и теперь набор данных готов для представления в отчетах. Необходимо сохранить результат работы в папку *data/clean*.

```
saveRDS(data, here("data", "clean", "cytomegalovirus.rds"))
```

Далее необходимо создать шаблон отчета *study_report_gt.Rmd* в папке *report_sources*. Данный отчет будет выводиться в формате HTML, поэтому его преамбула будет следующей.

```
---
title: "Study Report"
author: "Ivan Ivanov"
params:
  age.group: ""
  report.date: ""
output:
  html_document:
    toc: true
    theme: default
---
```

Следует опустить основную часть описания содержимого отчета — оно практически ничем не отличается от аналогичных файлов Rmarkdown: загрузка пакетов, задание параметров, выполнение скриптов, оформление результатов. Однако необходимо обратить внимание только запуск скрипта подготовки данных — он будет непосредственно вызываться при формировании отчета.

```
```{r prepare-data, include=FALSE}
source(here("scripts", "prepare-clean-data.R"))
```
```

Дополнительно можно создать еще один шаблон отчета, но уже для вывода в формат Word *study_report_ft.Rmd*. Он будет таким же, как и предыдущий, только будет использовать при выводе результатов пакет *flextable* и в преамбуле содержать следующее.

```
output:
  word_document:
    toc: true
```

Теперь структура папок фабрики отчетов выглядит следующим образом.

```
factory_overview()
├── README.md
├── RUN.R
├── data
│   ├── clean
│   │   └── cytomegalovirus.rds
│   └── raw
│       └── cytomegalovirus.csv
├── demo_factory.Rproj
├── factory_config
├── outputs
├── report_sources
│   ├── example_report.Rmd
│   ├── study_report_ft.Rmd
│   └── study_report_gt.Rmd
└── scripts
    └── prepare-clean-data.R
```

Посмотреть список доступных отчетов для генерации можно с помощью команды `list_reports()`.

```
list_reports()
[1] "example_report.Rmd" "study_report_ft.Rmd" "study_report_gt.Rmd"
```

После того, как определены все нужные скрипты и шаблоны отчетов, можно приступить к процессу компиляции отчетов. В фабрике отчетов “компиляция” отчета Rmarkdown означает, что скрипт .Rmd будет выполнен и будет создан выходной результат (как указано в преамбуле документа Rmarkdown - HTML, Word, PDF).

Для компиляции всех отчетов нужно выполнить команду `compile_reports()`.

```
compile_reports()
>>> Compiling report: example_report
>>> Compiling report: study_report_ft
>>> Compiling report: study_report_gt
All done!
```

Результат компиляции будет помещен в папку `output`, автоматически будет создана папка с меткой даты и времени. В этой папке будет сохранен сам отчет и все экспортанные файлы, созданные скриптом, а также сам файл Rmd (чтобы сохранить версию отчета на момент компиляции).

```
outputs
├── example_report
│   └── 2024-11-10_T22-19-56
│       ├── example_report.Rmd
│       └── example_report.html
```

```
study_report_ft
└── 2024-11-10_T22-19-56
    ├── study_report_ft.Rmd
    └── study_report_ft.docx
study_report_gt
└── 2024-11-10_T22-19-56
    ├── study_report_gt.Rmd
    └── study_report_gt.html
```

Такой поведение отличается от стандартного поведения компиляции файла Rmd, когда результат сохранялся рядом с исходным файлом. Фабрика же старается организовать хранение файлов для часто повторяемых генераций отчетов и сохранить версионность.

При необходимости можно выполнять компиляцию не всех отчетов, а только конкретных.

```
# Компиляция отчета по имени.
compile_reports(reports = "study_report_gt")
# Компиляция нескольких отчетов.
compile_reports(reports = c("study_report_gt", "example_report"))
```

Можно расположить группу отчетов в отдельные подпапки внутри папки *report_sources* и вызывать их следующей командой.

```
# Компиляция отчетов из папки.
compile_reports(reports = "example_folder/")
```

Отдельного внимания заслуживает возможность передать отчетам Rmarkdown параметры при компиляции. В этом случае для сохранения версионности следует помещать результат компиляции в отдельную подпапку.

```
compile_reports(reports = c("study_report_gt", "study_report_ft"),
                 params = list(age.group = "средний", report.date = "2024-11-07"),
                 subfolder = "средний возраст")
>>> Compiling report: study_report_gt
    - with parameters: age.group = средний, report.date = 2024-11-07
>>> Compiling report: study_report_ft
    - with parameters: age.group = средний, report.date = 2024-11-07
All done!
```

Структура файлов результата компиляции будет следующей.

```
outputs
├── study_report_ft
│   ├── 2024-11-10_T22-19-56
│   │   ├── study_report_ft.Rmd
│   │   └── study_report_ft.docx
│   └── средний возраст
│       └── 2024-11-10_T22-33-19
│           └── study_report_ft.Rmd
```

```
└── study_report_ft.docx
└── study_report_gt
    ├── 2024-11-10_T22-19-56
        ├── study_report_gt.Rmd
        └── study_report_gt.html
    └── средний возраст
        └── 2024-11-10_T22-33-19
            ├── study_report_gt.Rmd
            └── study_report_gt.html
```

В результате внутри папки *outputs* созданы подпапки для каждого отчета Rmd. В них созданы дополнительные подпапки для каждой уникальной компиляции. Эти папки имеют маркировку даты и времени. Внутри каждой папки с датой/временем находится скомпилированный отчет, а также скрипт Rmd.

Из дополнительных функций фабрики отчетов можно отметить показ всех результатов компиляций в компактном виде.

```
list_outputs()
[1] "example_report/2024-11-10_T22-19-56/example_report.html"
[2] "example_report/2024-11-10_T22-19-56/example_report.Rmd"
[3] "study_report_ft/2024-11-10_T22-19-56/study_report_ft.docx"
...
```

Еще одна полезная функция *list_deps()* — просмотр зависимостей всех скриптов и отчетов. Она позволяет вывести список всех пакетов, которые используются при подготовке отчета и при необходимости установить их с помощью команды *install_deps()*.

```
list_deps()
[1] "dplyr"      "here"       "rmarkdown" "fs"          "tidyverse" "flextable"
"officer"    "ggplot2"    "gt"         "knitr"      "base"
```

Таким образом, благодаря пакету *reportfactory* можно значительно упростить управление проектом с множеством Rmarkdown отчетов, позволяя компилировать сразу несколько отчетов и автоматически систематизировать результаты в папки с отметками времени. Встроенное управление зависимостями пакетов делает проекты легко переносимыми между устройствами, а готовность к интеграции с git-репозиториями облегчает командную работу и контроль версий.