array.r

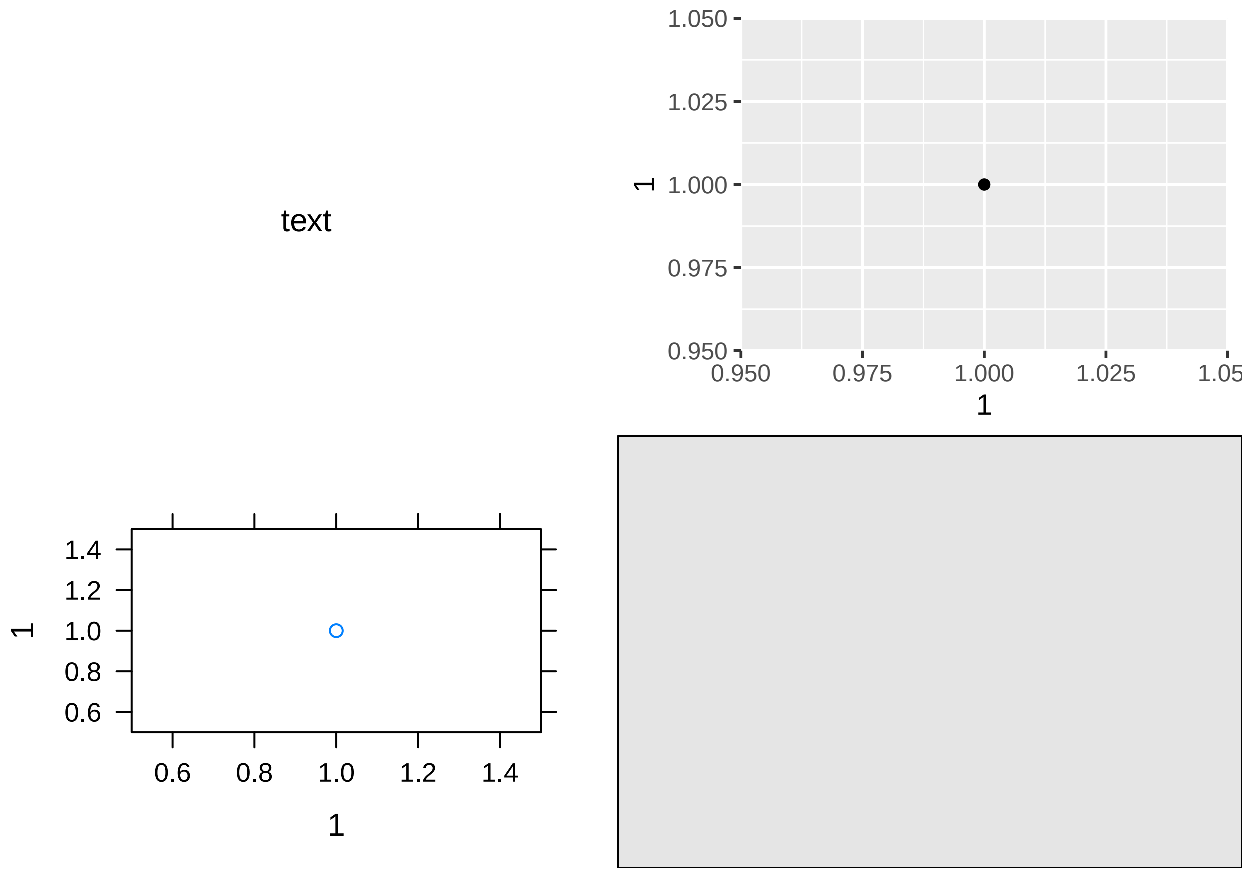
denis

2021-07-10

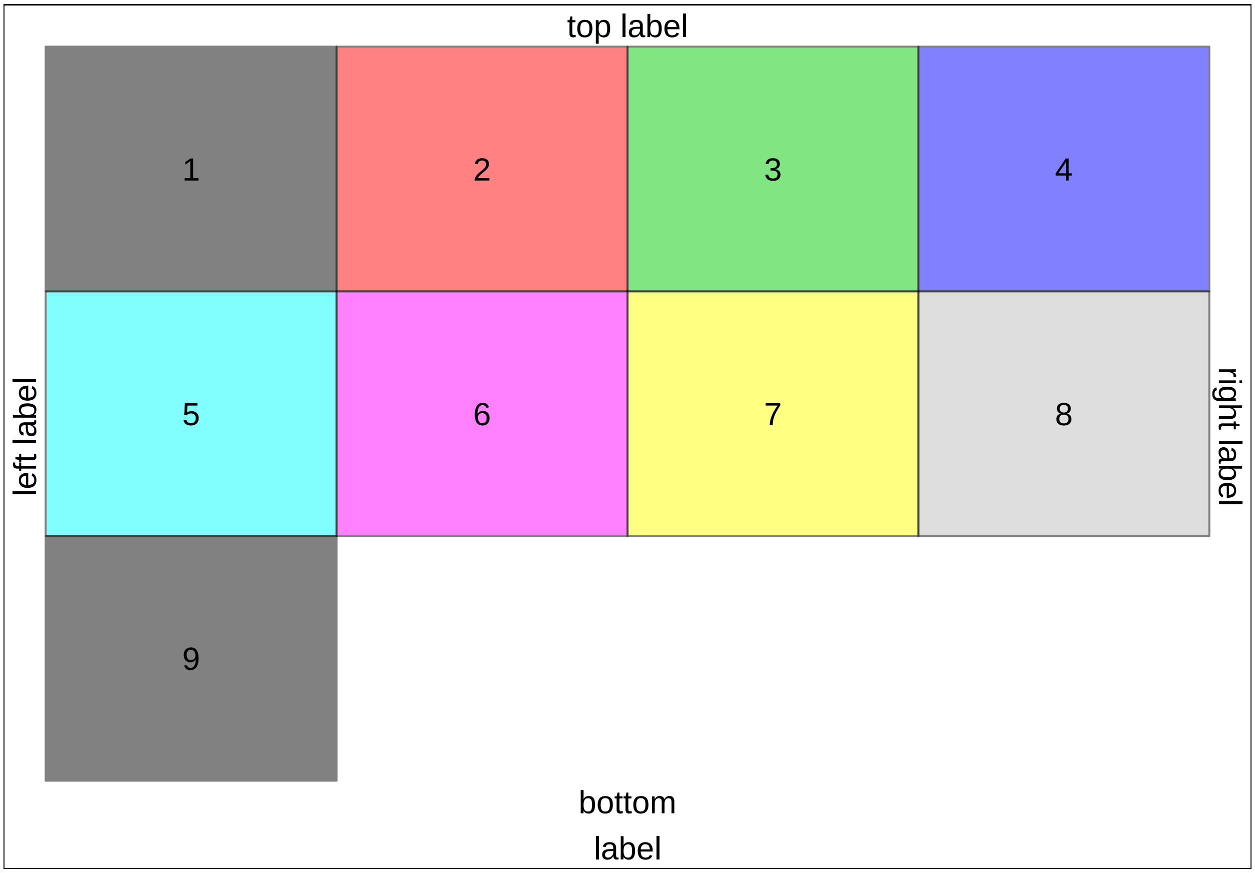
```
#!/usr/bin/r

# Many of the operations covered in the rust few chapters, especially the
# transformations and factorization in Chapter 5, are important because of
# their use in solving systems of linear equations, which will be discussed in
# Chapter 6; in computing eigenvectors, eigenvalues, and singular values, which
# will be discussed in Chapter 7; and in the applications in Chapter 9.

x <- 0:999
array(data = as.vector(x, mode = "any"), dim = length(x), dimnames = NULL)
```

```
##   [1]   0   1   2   3   4   5   6   7   8   9  10  11  12  13  14  15  16  17
##  [19]  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35
##  [37]  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53
##  [55]  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71
##  [73]  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89
##  [91]  90  91  92  93  94  95  96  97  98  99 100 101 102 103 104 105 106 107
## [109] 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125
## [127] 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143
## [145] 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161
## [163] 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179
## [181] 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197
## [199] 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215
## [217] 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233
## [235] 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251
## [253] 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269
## [271] 270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287
## [289] 288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305
## [307] 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323
## [325] 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340 341
## [343] 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359
## [361] 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377
## [379] 378 379 380 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395
## [397] 396 397 398 399 400 401 402 403 404 405 406 407 408 409 410 411 412 413
## [415] 414 415 416 417 418 419 420 421 422 423 424 425 426 427 428 429 430 431
## [433] 432 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448 449
## [451] 450 451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467
## [469] 468 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 484 485
## [487] 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 503
## [505] 504 505 506 507 508 509 510 511 512 513 514 515 516 517 518 519 520 521
## [523] 522 523 524 525 526 527 528 529 530 531 532 533 534 535 536 537 538 539
##  [541] 540 541 542 543 544 545 546 547 548 549 550 551 552 553 554 555 556 557
##  [559] 558 559 560 561 562 563 564 565 566 567 568 569 570 571 572 573 574 575
##  [577] 576 577 578 579 580 581 582 583 584 585 586 587 588 589 590 591 592 593
##  [595] 594 595 596 597 598 599 600 601 602 603 604 605 606 607 608 609 610 611
##  [613] 612 613 614 615 616 617 618 619 620 621 622 623 624 625 626 627 628 629
##  [631] 630 631 632 633 634 635 636 637 638 639 640 641 642 643 644 645 646 647
##  [649] 648 649 650 651 652 653 654 655 656 657 658 659 660 661 662 663 664 665
##  [667] 666 667 668 669 670 671 672 673 674 675 676 677 678 679 680 681 682 683
##  [685] 684 685 686 687 688 689 690 691 692 693 694 695 696 697 698 699 700 701
##  [703] 702 703 704 705 706 707 708 709 710 711 712 713 714 715 716 717 718 719
##  [721] 720 721 722 723 724 725 726 727 728 729 730 731 732 733 734 735 736 737
##  [739] 738 739 740 741 742 743 744 745 746 747 748 749 750 751 752 753 754 755
##  [757] 756 757 758 759 760 761 762 763 764 765 766 767 768 769 770 771 772 773
##  [775] 774 775 776 777 778 779 780 781 782 783 784 785 786 787 788 789 790 791
##  [793] 792 793 794 795 796 797 798 799 800 801 802 803 804 805 806 807 808 809
##  [811] 810 811 812 813 814 815 816 817 818 819 820 821 822 823 824 825 826 827
##  [829] 828 829 830 831 832 833 834 835 836 837 838 839 840 841 842 843 844 845
##  [847] 846 847 848 849 850 851 852 853 854 855 856 857 858 859 860 861 862 863
##  [865] 864 865 866 867 868 869 870 871 872 873 874 875 876 877 878 879 880 881
##  [883] 882 883 884 885 886 887 888 889 890 891 892 893 894 895 896 897 898 899
##  [901] 900 901 902 903 904 905 906 907 908 909 910 911 912 913 914 915 916 917
##  [919] 918 919 920 921 922 923 924 925 926 927 928 929 930 931 932 933 934 935
##  [937] 936 937 938 939 940 941 942 943 944 945 946 947 948 949 950 951 952 953
##  [955] 954 955 956 957 958 959 960 961 962 963 964 965 966 967 968 969 970 971
##  [973] 972 973 974 975 976 977 978 979 980 981 982 983 984 985 986 987 988 989
##  [991] 990 991 992 993 994 995 996 997 998 999
```

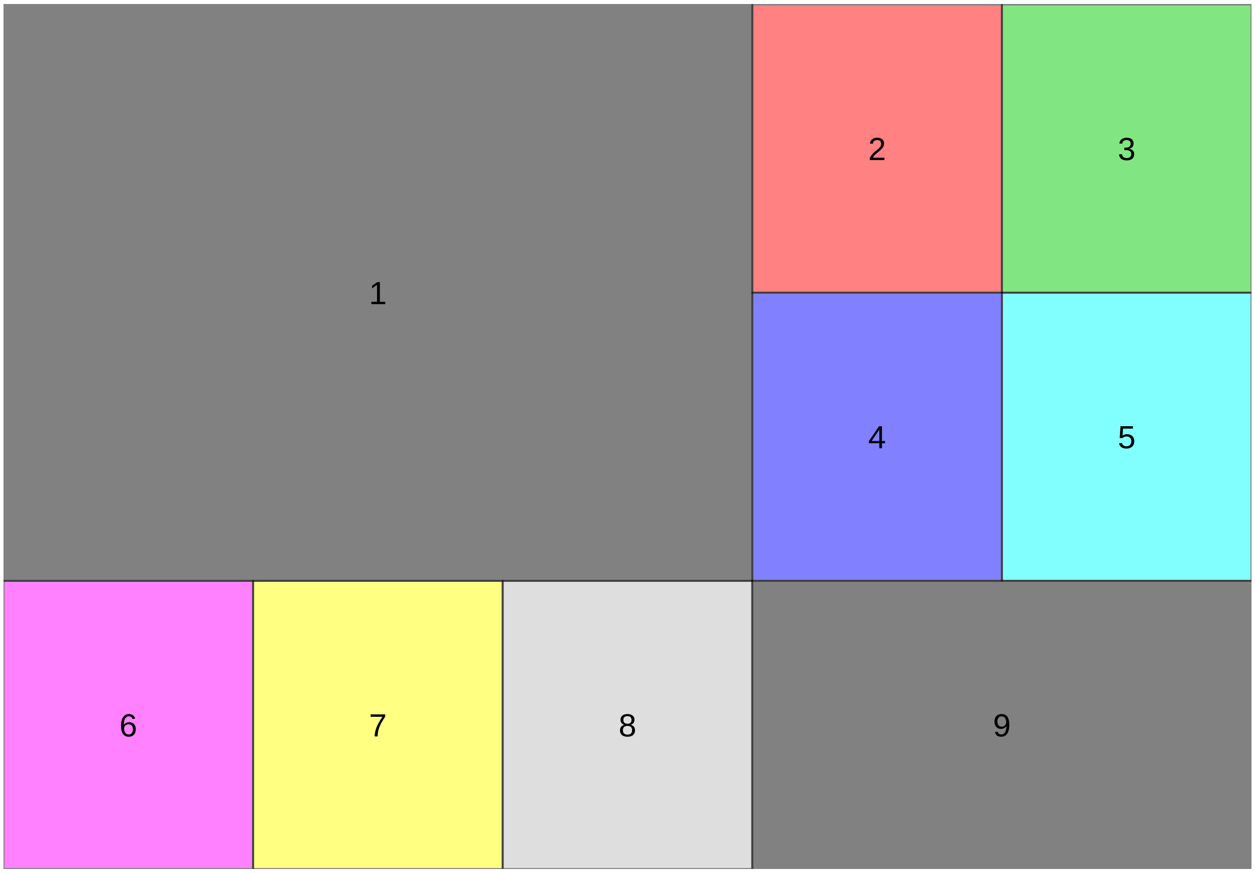
```
# Throughout the rust few chapters, we emphasize the facts that are import-
# tat in statistical applications. We also occasionally refer to relevant comps-
# national issues, although computational details are addressed specifically in
# Part III.
library(gridExtra)
library(grid)
library(ggplot2)
library(lattice)

p <- qplot(1,1)
p2 <- xyplot(1~1)
r <- rectGrob(gp=gpar(fill="grey90"))
t <- textGrob("text")
grid.arrange(t, p, p2, r, ncol=2)
```



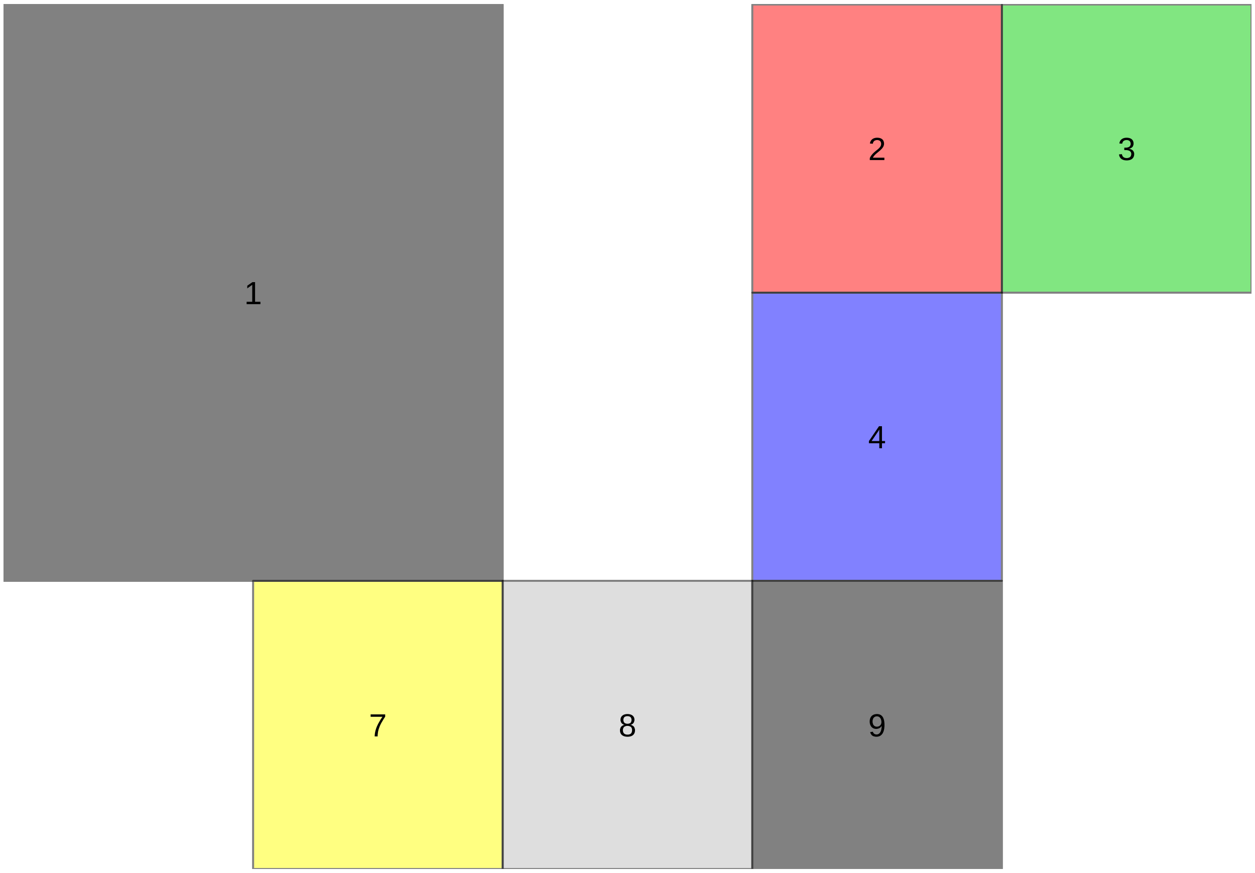
```
gs <- lapply(1:9, function(ii)
grobTree(rectGrob(gp=gpar(fill=ii, alpha=0.5)), textGrob(ii)))
grid.arrange(grobs=gs, ncol=4, top="top label",
               bottom="bottom\nlabel",
               left="left label", right="right label")
grid.rect(gp=gpar(fill=NA))
```



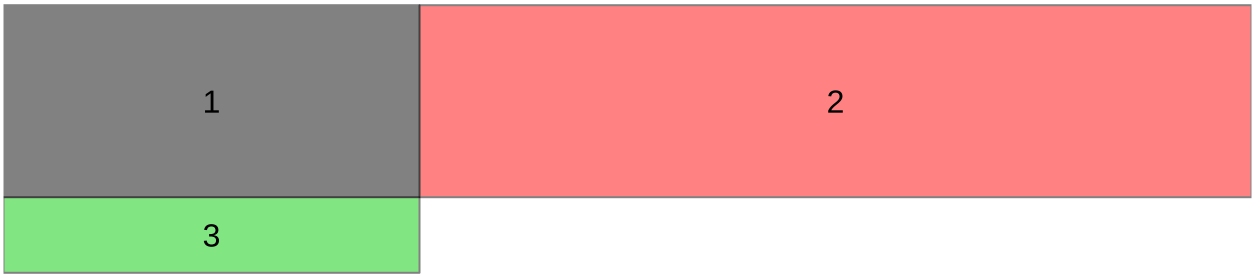
```r
# Complex layouts
# We can provide a matrix defining the layout,
lay <- rbind(c(1,1,1,2,3),
             c(1,1,1,4,5),
             c(6,7,8,9,9))

grid.arrange(grobs = gs, layout_matrix = lay)
```



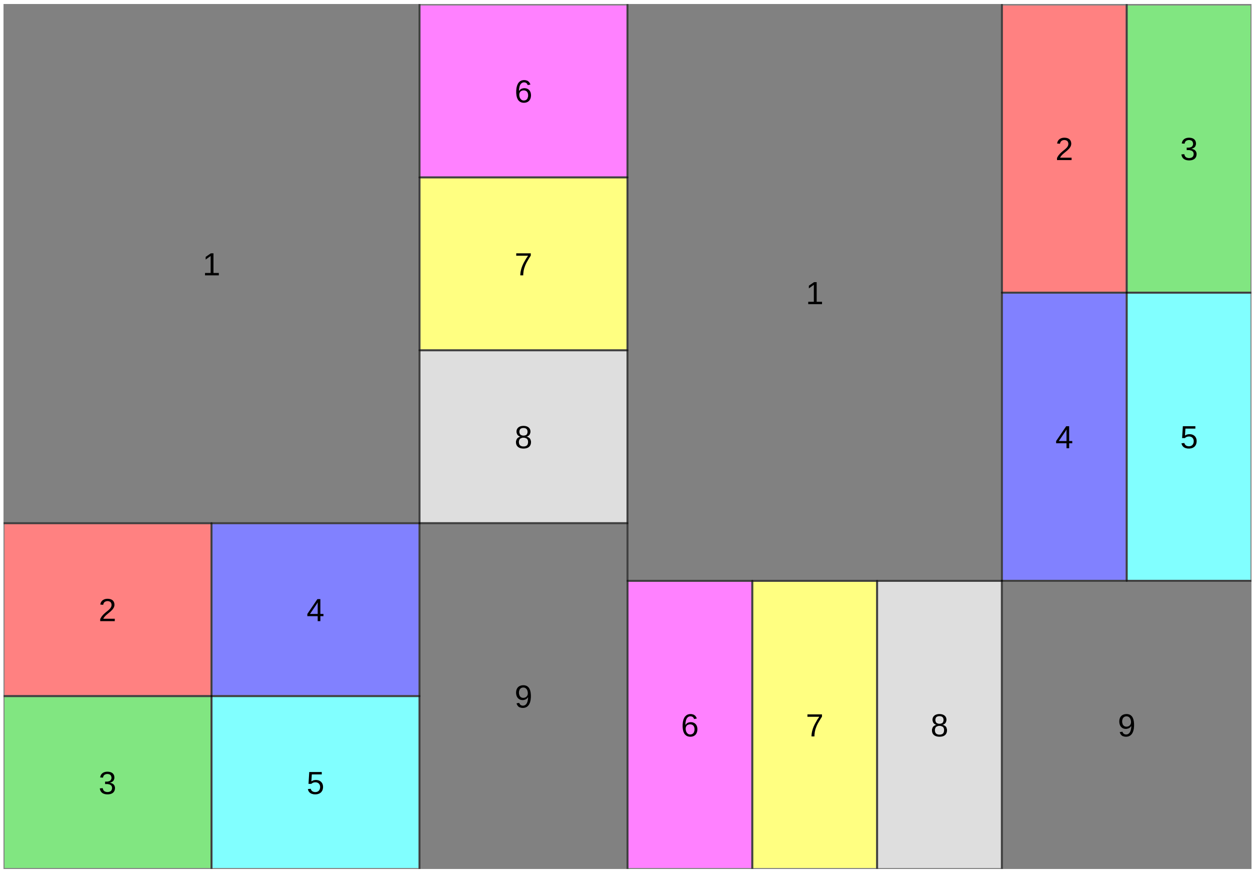
```
# The layout itself may contain holes, but note that for any given grob index
# the region must be simply connected (no hole),

hlay <- rbind(c(1,1,NA,2,3),
              c(1,1,NA,4,NA),
              c(NA,7,8,9,NA))

select_grobs <- function(lay){
  id <- unique(c(t(lay)))
  id[!is.na(id)]
}

grid.arrange(grobs=gs[select_grobs(hlay)], layout_matrix=hlay)
```



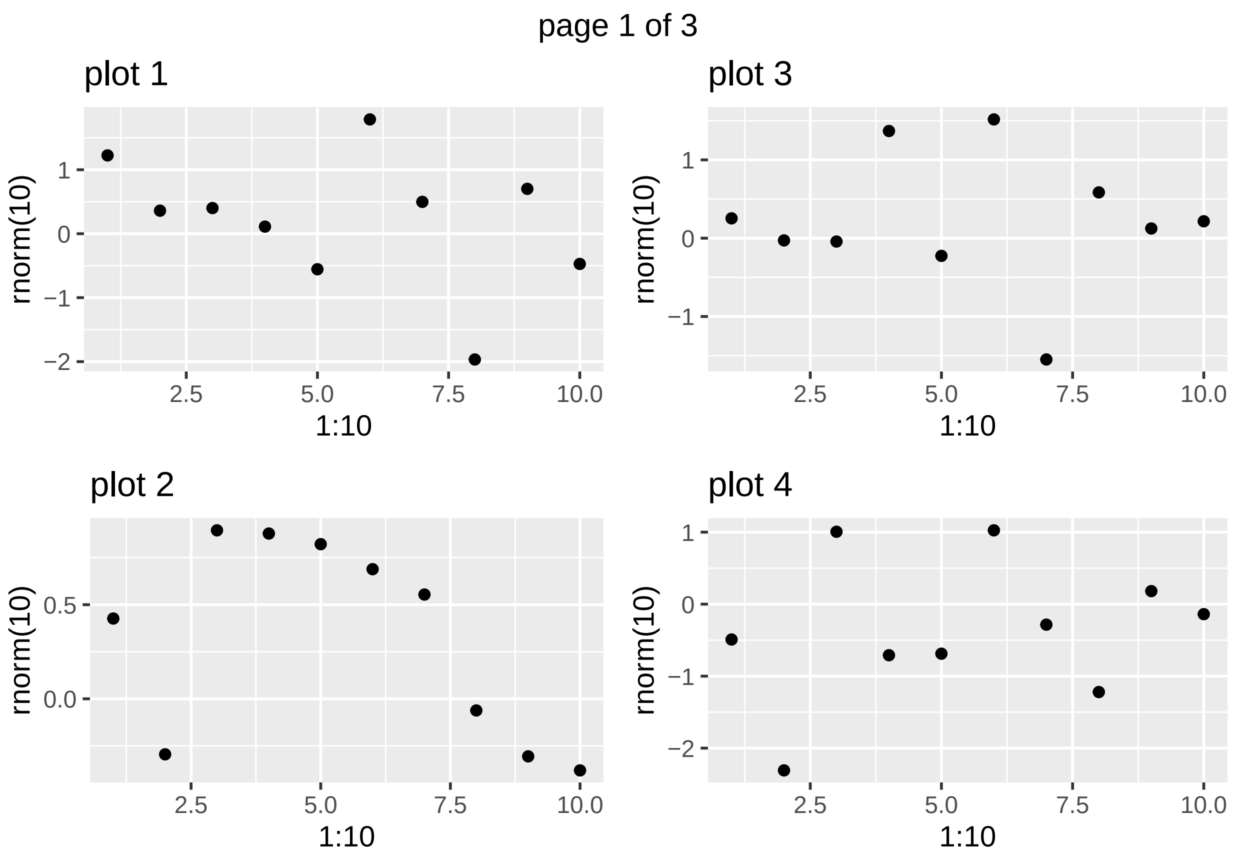
```
# All cells are of equal size by default, but users may pass explicitly widths
# and/or heights in any valid grid units, or as relative numbers (interpreted
# as null),
grid.arrange(grobs=gs[1:3], ncol = 2, widths = 1:2, heights=unit(c(1,10), c("in", "mm")))
```



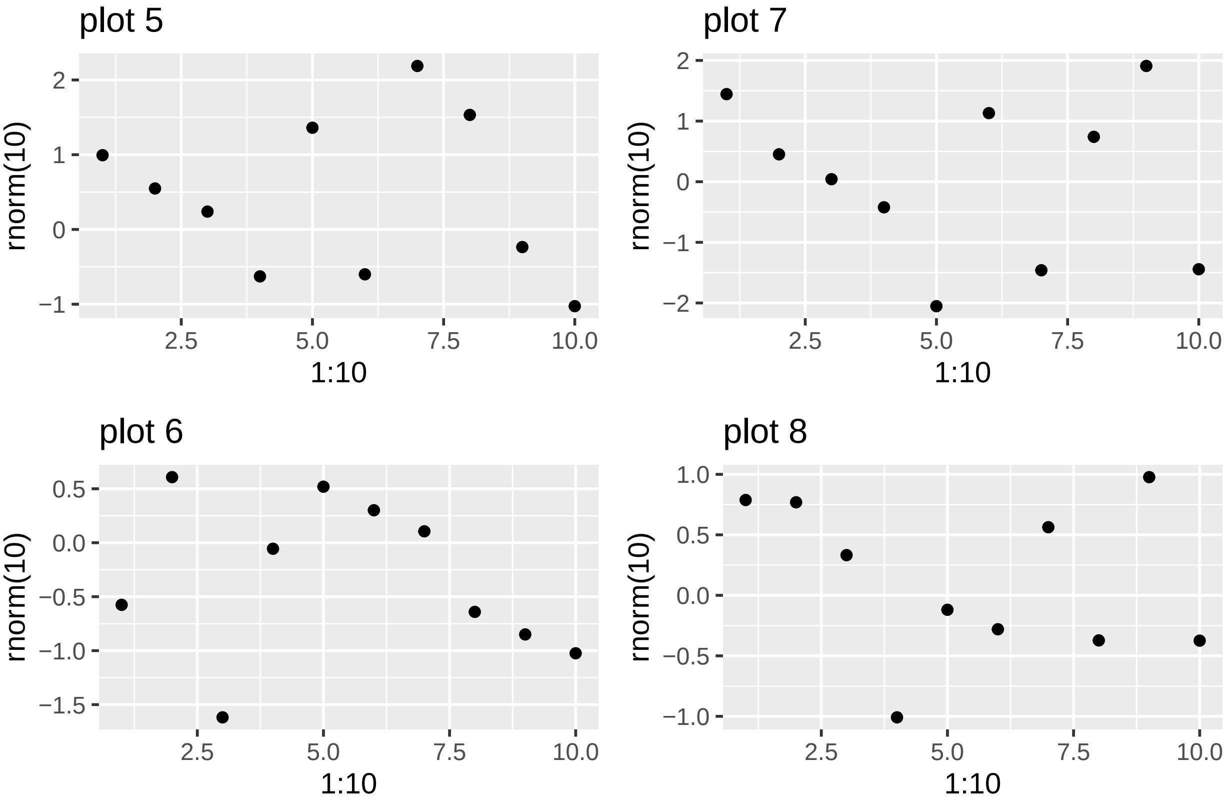
```r
# Nested layouts with arrangeGrob
# The grid.arrange() function draws on the device; for more complex layouts, we
# may want to store the gable and combine it with other objects, e.g. forming
# nested layouts. To this end, use arrangeGrob(),
g1 <- arrangeGrob(grobs = gs, layout_matrix = t(lay))
g2 <- arrangeGrob(grobs = gs, layout_matrix = lay)
grid.arrange(g1, g2, ncol=2)
```



```r
# Multiple pages output
# Finally, we may want to place grobs on multiple pages; the marrangeGrob()
# function provides a convenient interface for this, also compatible with
# ggsave().

set.seed(123)
pl <- lapply(1:11, function(.x)
      qplot(1:10, rnorm(10), main=paste("plot", .x)))

ml <- marrangeGrob(pl, nrow = 2, ncol = 2)

## non-interactive use, multipage pdf
## ggsave("multipage.pdf", ml)
## interactive use; calling `dev.new` multiple times
ml
```


page 1 of 3
plot 1
rnorm(10)
1
0
−1
−2
2.5
5.0
7.5
10.0
1:10
plot 3
rnorm(10)
1
0
−1
2.5
5.0
7.5
10.0
1:10
plot 2
rnorm(10)
0.5
0.0
2.5
5.0
7.5
10.0
1:10
plot 4
rnorm(10)
1
0
−1
−2
2.5
5.0
7.5
10.0
1:10

page 2 of 3



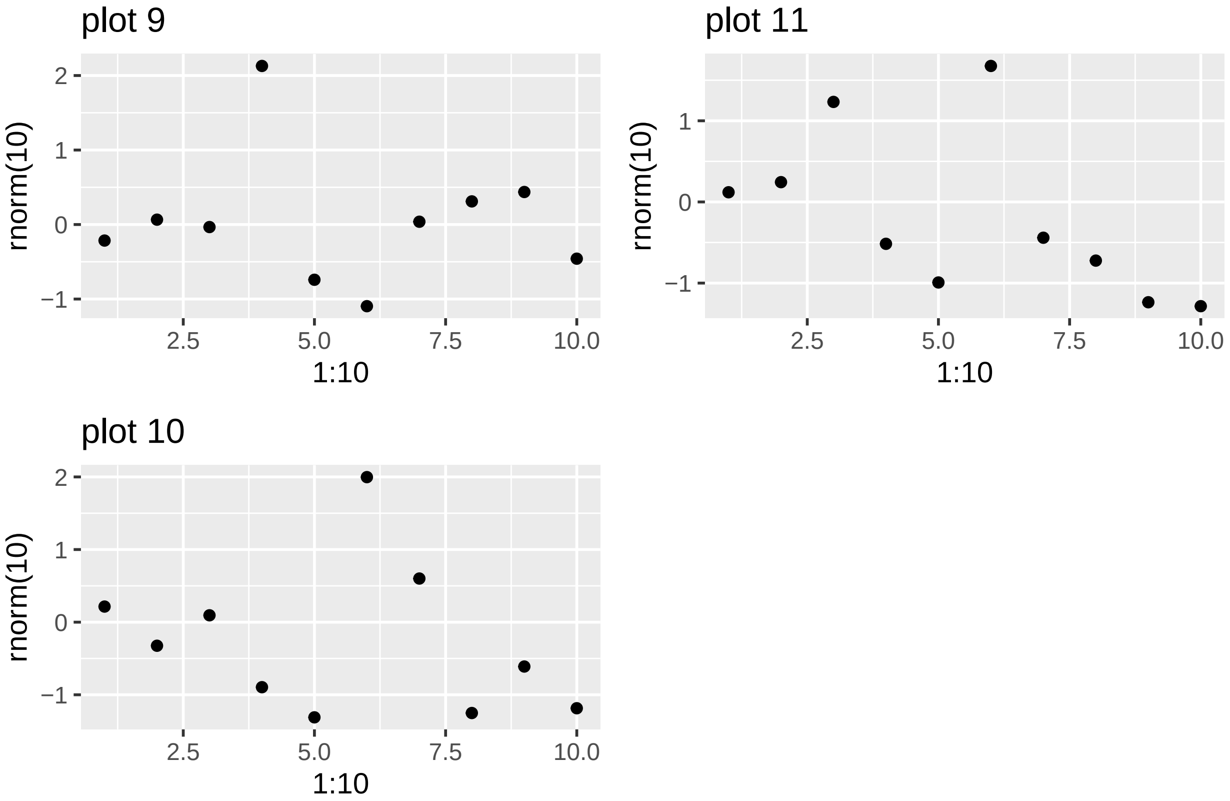
plot 5

plot 7

plot 6

plot 8

page 3 of 3


plot 9
rnorm(10)
2
1
0
−1
2.5
5.0
7.5
10.0
1:10
plot 11
rnorm(10)
1
0
−1
2.5
5.0
7.5
10.0
1:10
plot 10
rnorm(10)
2
1
0
−1
2.5
5.0
7.5
10.0
1:10