Topic 7p: Percentiles

To look at percentiles it is helpful to have a large data set.

gnrnd5(41583023104, 31000444)

454	446	426	455	464	530	446	447	513	395	441	433	438	450	416	412	473	437	411	451
420	414	461	496	395	392	465	411	413	408	430	387	508	383	405	415	435	442	461	433
401	431	422	439	424	419	438	444	397	421	502	425	458	478	454	418	460	417	417	474
440	445	437	438	458	427	444	398	444	461	496	476	467	443	519	436	407	446	421	485
458	443	453	434	487	429	449	408	440	391	451	403	412	400	464	462	505	433	446	480
456	486	436	451	449	470	489	431	432	481	458	467	429	461	486	430	436	480	464	438
452	414	459	452	491	454	395	447	461	501	474	455	454	474	478	409	472	490	477	431
445	411	401	424	450	393	405	438	470	404	467	461	433	471	444	508	470	406	486	456
459	462	417	518	460	379	505	419	443	433	415	427	450	484	447	488	477	414	473	413
477	411	459	407	473	445	446	437	414	444	451	455	385	446	495	470	478	473	427	422
471	396	456	393	476	438	456	408	428	429	402	499	428	455	440	430	428	473	454	463
434	471	412	446	464	456	407	418	446	478	410	489								

But what we really need is to have that data sorted.

```
  [1] 379 383 385 387 391 392 393 393 395 395 395 396 397 398 400
 [16] 401 401 402 403 404 405 405 406 407 407 407 408 408 408 409
 [31] 410 411 411 411 411 412 412 412 413 413 414 414 414 414 415
 [46] 415 416 417 417 417 418 418 419 419 420 421 421 422 422 424
 [61] 424 425 426 427 427 427 428 428 428 429 429 429 430 430 430
 [76] 431 431 431 432 433 433 433 433 433 434 434 435 436 436 436
 [91] 437 437 437 438 438 438 438 438 438 439 440 440 440 441 442
[106] 443 443 443 444 444 444 444 444 445 445 445 446 446 446 446
[121] 446 446 446 446 447 447 447 449 449 450 450 450 451 451 451
[136] 451 452 452 453 454 454 454 454 454 455 455 455 455 456 456
[151] 456 456 456 458 458 458 458 459 459 459 460 460 461 461 461
[166] 461 461 461 462 462 463 464 464 464 464 465 467 467 467 470
[181] 470 470 470 471 471 471 472 473 473 473 473 473 474 474 474
[196] 476 476 477 477 477 478 478 478 478 480 480 481 484 485 486
[211] 486 486 487 488 489 489 490 491 495 496 496 499 501 502 505
[226] 505 508 508 513 518 519 530
```

Then the 41st percentile is the value in the sorted list that has 41% of the values less than it. Since there are 232 values in the sorted list, find 0.41*232. That is 95.12. Round that up to 96. Find item 96 in the list. That is one of the 438's. So, the 41st percentile is 438.

To find the 95th percentile, find 0.95*232. That is 220.4. Find the 221st item. That is 496. So, the 95th percentile is 496.

To find the 43rd percentile, find 0.43*232. That is 99.76. Find the 100th item in the sorted list. That is 439. So, the 43rd percentile is 439. That is the best we can do by hand. Now, look to see how we can do this in R.

R uses the quantile() function to find percentiles. [Quantiles are expressed as decimals, as in 0.41 while percentiles are given in that form, as in 41%.] To get the 41st percentile we use the command quantile(L1, 0.41).

```
> quantile(L1, 0.41)
41%
438
```

This is the same answer that we got by hand.

To find the 95th percentile we use the command quantile(L1, 0.95).

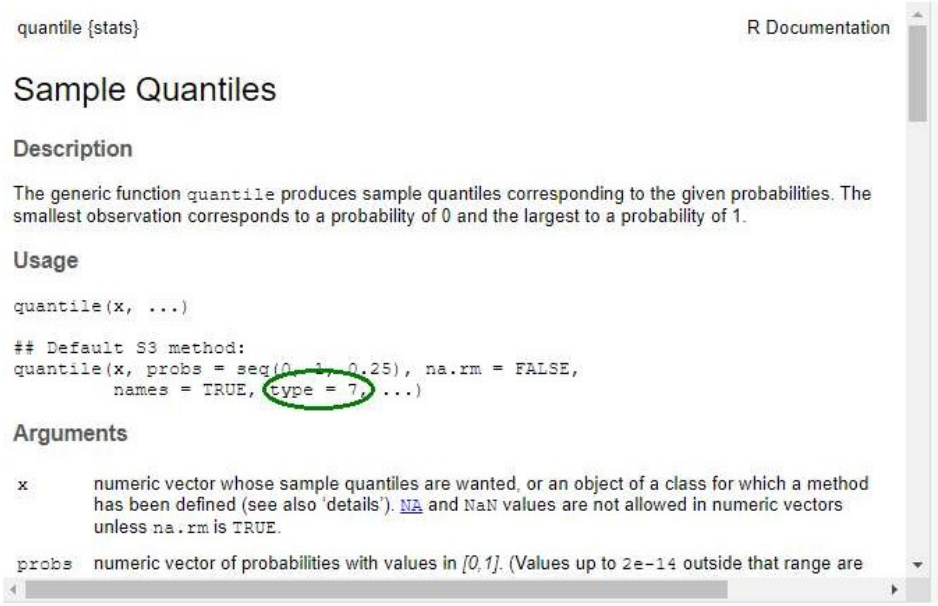
```
> quantile(L1, 0.95)
95%
496
```

Again this matches our earlier answer. To find the 43rd percentile we use the command quantile(L1, 0.43).

```
> quantile(L1, 0.43)
   43%
439.33
```

This is a different answer than we got by hand. Why? It turns out that there are at least 9 different ways to compute quantiles. In reality, nobody is going to care if we report the answer as 439 or 439.33. For this class we will always take the R answer.

We can take a moment to look at the help information about quantile(). To do this we use the command ?quantile and then run that. The result is a help page in the lower right pane of our screen.



quantile {stats} R Documentation

Sample Quantiles

Description

The generic function `quantile` produces sample quantiles corresponding to the given probabilities. The smallest observation corresponds to a probability of 0 and the largest to a probability of 1.

Usage

```
quantile(x, ...)

## Default S3 method:
quantile(x, probs = seq(0, 1, 0.25), na.rm = FALSE,
         names = TRUE, type = 7, ...)
```

Arguments

`x`	numeric vector whose sample quantiles are wanted, or an object of a class for which a method has been defined (see also 'details'). `NA` and `NaN` values are not allowed in numeric vectors unless `na.rm` is `TRUE`.
`probs`	numeric vector of probabilities with values in *[0,1]*. (Values up to `2e-14` outside that range are

Notice that we have a default of type=7. We can scroll down to find more information.

`type`	an integer between 1 and 9 selecting one of the nine quantile algorithms detailed below to be used.

And then further we find:

Type 6

m = p. *p[k] = k / (n + 1)*. Thus *p[k] = E[F(x[k])]*. This is used by Minitab and by SPSS.

Type 7

m = 1-p. *p[k] = (k - 1) / (n - 1)*. In this case, *p[k] = mode[F(x[k])]*. This is used by S.

Type 8

m = (p+1)/3. *p[k] = (k - 1/3) / (n + 1/3)*. Then *p[k] =~ median[F(x[k])]*. The resulting quantile estimates are approximately median-unbiased regardless of the distribution of `x`.

All of this may just be too much information, but it does give us a hint as to why different systems give slightly different answers to the same problem.

Just a bit more on percentiles (i.e., quantiles). We can get a whole set of percentiles by giving the quantile function a list of quantile values.

```
> # and we can give quantile() a whole list of
> # percentiles to find
> quantile( L1, c(0.20, 0.33, 0.14, 0.78))
   20%    33%    14%    78%
28.560 33.100 26.400 52.772
>      # or even a sequence of values
> quantile( L1, seq(0.60,0.95,0.05))
  60%   65%   70%   75%   80%   85%   90%   95%
47.92 49.41 50.68 52.20 53.30 54.69 55.76 57.60
```

And we can go backwards in this process. To find what percentile is a particular value we can find, in the sorted list, the position of that value (or the value just less than that value), and then express the ratio of that position to the length of the list as a percent.

If we have the list of sorted valuees we can figure out the position. Going back to our sorted list we see that 472 is in position 187 so we compute that 472 is in the (187/232)*100 percentile, that is the 80th percentile, meaning that 80 % of the values are less than 472.

If we are looking for a value that is not in the list then we find the position of the highest value less than our value. The percentile for 494 is (218/232)*100=93.96 or the 93rd percentile because the highest value less than the missing 494 is 491 and it is in position 218.

For values that are in the list, R does give us a way to find the position of the value. Note the double equal sign in the following statement.

```
>      # If we want to know what percentile is the value
>      # 487 we find 487 in the sorted data and then get
>      # its position
> which( L2==487)
[1] 213
>      # then find the percent that index is of the
>      # total size of the data and then round down,
>      # but we never go over 99%
> 213/232*100
[1] 91.81034
```