

## 2.12 Appendix: Descriptive Statistics with R

by EV Nordheim, MK Clayton & BS Yandell, September 7, 2003

Virtually every statistical computing package can present graphical displays of data and report summary statistics. We demonstrate how to use R to produce some of the graphs and summary statistics discussed in Chapter 2.

Consider first the red pine seedling data. We can enter the data in R as

```
> redpine = c(42, 23, 43, 34, 49, 56, 31, 47, 61, 54, 46, 34, 26)
> redpine
[1] 42 23 43 34 49 56 31 47 61 54 46 34 26
> length(redpine)
[1] 13
```

The first command line creates the `redpine` object, while the second line prints it so that we can check it. It never hurts to check the length of an object as well, which is just the sample size  $n$ . We can produce a stem and leaf display using the `stem` command in R.

```
> stem(redpine)
The decimal point is 1 digit(s) to the right of the |

 2 | 36
 3 | 144
 4 | 23679
 5 | 46
 6 | 1
```

Notice that there are 5 categories for these 13 numbers, with stems for the 10s digit and leaves for the 1s digit. The stem and leaf plot can be scaled to have more stems by changing the `scale` option:

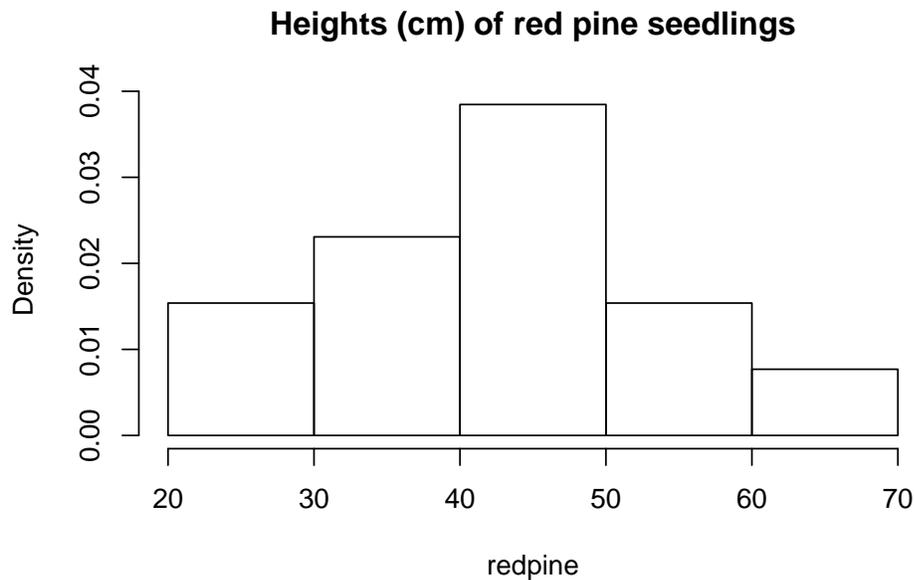
```
> stem(redpine, scale = 2)
The decimal point is 1 digit(s) to the right of the |

 2 | 3
 2 | 6
 3 | 144
 3 |
 4 | 23
 4 | 679
 5 | 4
 5 | 6
 6 | 1
```

The `stem` command always produces ordered stem and leaf displays.

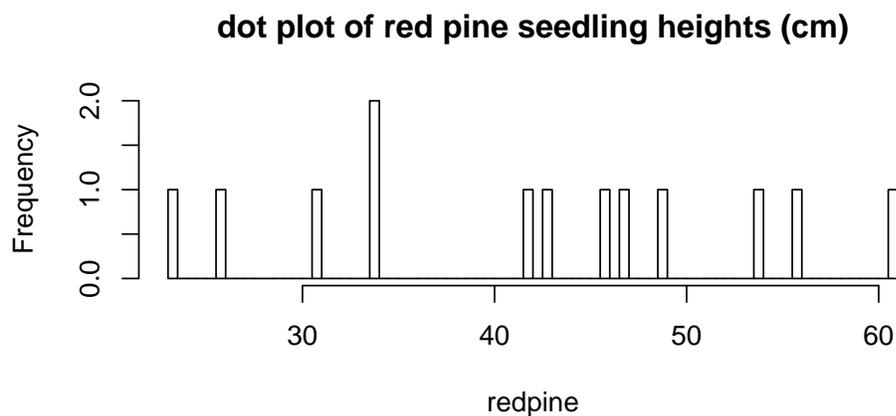
The command for constructing histograms, `hist(redpine)`, can take arguments that control the specific form of the display. As we noted, this is particularly important for histograms. Here is a *density scaled* histogram (`prob=TRUE`) of the red pine data:

```
> hist(redpine, prob = TRUE, main = "Heights (cm) of red pine seedlings")
```



Currently there is no `dotplot` command in R. However, you can get essentially the same thing by using a large value for the `breaks` argument to the `hist` command. We leave further details to the reader.

```
> hist(redpine, breaks = 100,  
+ main = "dot plot of red pine seedling heights (cm)")
```



The easiest method for obtaining useful numerical summary statistics is to use the `summary` command, which includes the commonly used measure of location (quartiles, median and mean) as well as the minimum and maximum values.

```
> summary(redpine)
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
23	34	43	42	49	61

The R commands `mean(redpine)` and `median(redpine)` give you the corresponding values on their own.

The `summary` command includes some raw ingredients for measures of spread. There are separate commands for the SD, IQR and range:

```
> sd(redpine)
```

```
[1] 11.75443
```

```
> IQR(redpine)
```

```
[1] 15
```

```
> diff(range(redpine))
```

```
[1] 38
```

The `range` command returns the smallest and largest values, and their difference is calculated by the `diff` command. The interquartile range (IQR) is the difference between the third and first quartiles. Recall that the variance is the square of the standard deviation, which you can verify:

```
> var(redpine)
```

```
[1] 138.1667
```

```
> sd(redpine)^2
```

```
[1] 138.1667
```

You can reorganize these measures of spread by hand using your favorite word processor, or you can use R to help. For instance,

```
> c(SD = sd(redpine), IQR = IQR(redpine), Range = diff(range(redpine)))
```

	SD	IQR	Range
	11.75443	15.00000	38.00000

Let us briefly reconsider the data set giving the watershed areas of 18 lakes in Northern Wisconsin. Here is the stem and leaf display.

```
> watershed = c(0.3, 0.3, 0.4, 0.5, 0.5, 1.3, 1.6, 1.8, 2.6, 2.6,
+ 2.6, 4.7, 9.1, 9.1, 12.9, 19.4, 33.7, 176.1)
> stem(watershed)
```

The decimal point is 2 digit(s) to the right of the |

```
0 | 000000000000011123
0 |
1 |
1 | 8
```

R does some rounding of the data values so, for example, all the observations with area below 10 km<sup>2</sup> are represented in the display with stem “0” and leaf “0”. As noted earlier in this chapter, these data are highly skewed.

It is quite straightforward to transform these data, say with the logarithm (to the base 10) using the `log10` command. (Logarithms to the base  $e$  are produced using the `ln` command.) We proceed as follows.

```
> logshed = log10(watershed)
> stem(logshed)
```

The decimal point is at the |

```
-0 | 55433
0 | 1234447
1 | 00135
2 | 2
```

Alternatively, we could have done this in one step:

```
> stem(log10(watershed))
```

The decimal point is at the |

```
-0 | 55433
0 | 1234447
1 | 00135
2 | 2
```