Intermediate R programming exercise solutions

Kim Dill-McFarland

version October 16, 2018

Setup

We will be working with the same data and packages as in the notes and main.R files.

```
library(tidyverse)
library(lmerTest)
library(devtools)
library(roxygen2)
```

```
raw_data <- read_csv(file="data/Saanich_Data.csv",
                     col_names=TRUE,
                     na=c("", "NA", "NAN", "ND"))

dat <-
  raw_data %>%
  select(Cruise, Date, Depth,
         Temperature, Salinity, Density,
         WS_O2, WS_NO3, WS_H2S) %>%
  filter(Date >= "2008-02-01") %>%
  rename(O2=WS_O2, NO3=WS_NO3, H2S=WS_H2S) %>%
  mutate(Depth=Depth*1000)
```

S3 objects

Vectors

1. Assign x the value `"a"`. What are its class and mode?

```
x <- "a"
class(x)
```

```
## [1] "character"
```

```
mode(x)
```

```
## [1] "character"
```

2. Give it dimensions `c(1,1)`. What are its class and mode?

```
dim(x) <- c(1,1)
class(x)
```

```
## [1] "matrix"
```

```
mode(x)
```

```
## [1] "character"
```

Data objects

1. Calculate a summary table of `dat`. What are its class and attributes?

```
sum <- summary(dat)
class(sum)
```

```
## [1] "table"
```

```
attributes(sum)
```

```
## $dim
## [1] 7 9
##
## $dimnames
## $dimnames[[1]]
## [1] "" "" "" "" "" "" ""
##
## $dimnames[[2]]
## [1] "    Cruise"   "     Date"    "    Depth"    " Temperature"
## [5] "   Salinity"  "   Density"   "      O2"     "     NO3"
## [9] "     H2S"
##
##
## $class
## [1] "table"
```

2. Read in the raw data table `Saanich_Data.csv` using the base R function `read.table`. What are this object's class and attributes? Are they any different from the object create when we used `read_csv` to read in the same data?

```
dat2 <- read.table("data/Saanich_Data.csv", sep=",")
class(dat2)
```

```
## [1] "data.frame"
```

```
attributes(dat2)
```

```
## $names
##  [1] "V1"  "V2"  "V3"  "V4"  "V5"  "V6"  "V7"  "V8"  "V9"  "V10" "V11"
## [12] "V12" "V13" "V14" "V15" "V16" "V17" "V18" "V19" "V20" "V21" "V22"
## [23] "V23" "V24" "V25" "V26" "V27" "V28" "V29"
## 
## $class
## [1] "data.frame"
## 
## $row.names
##    [1]    1    2    3    4    5    6    7    8    9   10   11   12   13
##   [14]   14   15   16   17   18   19   20   21   22   23   24   25   26
##   [27]   27   28   29   30   31   32   33   34   35   36   37   38   39
##   [40]   40   41   42   43   44   45   46   47   48   49   50   51   52
##   [53]   53   54   55   56   57   58   59   60   61   62   63   64   65
##   [66]   66   67   68   69   70   71   72   73   74   75   76   77   78
##   [79]   79   80   81   82   83   84   85   86   87   88   89   90   91
##   [92]   92   93   94   95   96   97   98   99  100  101  102  103  104
##  [105]  105  106  107  108  109  110  111  112  113  114  115  116  117
##  [118]  118  119  120  121  122  123  124  125  126  127  128  129  130
##  [131]  131  132  133  134  135  136  137  138  139  140  141  142  143
##  [144]  144  145  146  147  148  149  150  151  152  153  154  155  156
##  [157]  157  158  159  160  161  162  163  164  165  166  167  168  169
##  [170]  170  171  172  173  174  175  176  177  178  179  180  181  182
##  [183]  183  184  185  186  187  188  189  190  191  192  193  194  195
##  [196]  196  197  198  199  200  201  202  203  204  205  206  207  208
##  [209]  209  210  211  212  213  214  215  216  217  218  219  220  221
##  [222]  222  223  224  225  226  227  228  229  230  231  232  233  234
##  [235]  235  236  237  238  239  240  241  242  243  244  245  246  247
##  [248]  248  249  250  251  252  253  254  255  256  257  258  259  260
##  [261]  261  262  263  264  265  266  267  268  269  270  271  272  273
##  [274]  274  275  276  277  278  279  280  281  282  283  284  285  286
##  [287]  287  288  289  290  291  292  293  294  295  296  297  298  299
##  [300]  300  301  302  303  304  305  306  307  308  309  310  311  312
##  [313]  313  314  315  316  317  318  319  320  321  322  323  324  325
##  [326]  326  327  328  329  330  331  332  333  334  335  336  337  338
##  [339]  339  340  341  342  343  344  345  346  347  348  349  350  351
##  [352]  352  353  354  355  356  357  358  359  360  361  362  363  364
##  [365]  365  366  367  368  369  370  371  372  373  374  375  376  377
##  [378]  378  379  380  381  382  383  384  385  386  387  388  389  390
##  [391]  391  392  393  394  395  396  397  398  399  400  401  402  403
##  [404]  404  405  406  407  408  409  410  411  412  413  414  415  416
##  [417]  417  418  419  420  421  422  423  424  425  426  427  428  429
##  [430]  430  431  432  433  434  435  436  437  438  439  440  441  442
##  [443]  443  444  445  446  447  448  449  450  451  452  453  454  455
##  [456]  456  457  458  459  460  461  462  463  464  465  466  467  468
##  [469]  469  470  471  472  473  474  475  476  477  478  479  480  481
##  [482]  482  483  484  485  486  487  488  489  490  491  492  493  494
##  [495]  495  496  497  498  499  500  501  502  503  504  505  506  507
##  [508]  508  509  510  511  512  513  514  515  516  517  518  519  520
##  [521]  521  522  523  524  525  526  527  528  529  530  531  532  533
##  [534]  534  535  536  537  538  539  540  541  542  543  544  545  546
##  [547]  547  548  549  550  551  552  553  554  555  556  557  558  559
##  [560]  560  561  562  563  564  565  566  567  568  569  570  571  572
##  [573]  573  574  575  576  577  578  579  580  581  582  583  584  585
##  [586]  586  587  588  589  590  591  592  593  594  595  596  597  598
##  [599]  599  600  601  602  603  604  605  606  607  608  609  610  611
##  [612]  612  613  614  615  616  617  618  619  620  621  622  623  624
##  [625]  625  626  627  628  629  630  631  632  633  634  635  636  637
##  [638]  638  639  640  641  642  643  644  645  646  647  648  649  650
##  [651]  651  652  653  654  655  656  657  658  659  660  661  662  663
##  [664]  664  665  666  667  668  669  670  671  672  673  674  675  676
##  [677]  677  678  679  680  681  682  683  684  685  686  687  688  689
##  [690]  690  691  692  693  694  695  696  697  698  699  700  701  702
##  [703]  703  704  705  706  707  708  709  710  711  712  713  714  715
##  [716]  716  717  718  719  720  721  722  723  724  725  726  727  728
##  [729]  729  730  731  732  733  734  735  736  737  738  739  740  741
##  [742]  742  743  744  745  746  747  748  749  750  751  752  753  754
##  [755]  755  756  757  758  759  760  761  762  763  764  765  766  767
##  [768]  768  769  770  771  772  773  774  775  776  777  778  779  780
##  [781]  781  782  783  784  785  786  787  788  789  790  791  792  793
##  [794]  794  795  796  797  798  799  800  801  802  803  804  805  806
##  [807]  807  808  809  810  811  812  813  814  815  816  817  818  819
##  [820]  820  821  822  823  824  825  826  827  828  829  830  831  832
##  [833]  833  834  835  836  837  838  839  840  841  842  843  844  845
##  [846]  846  847  848  849  850  851  852  853  854  855  856  857  858
##  [859]  859  860  861  862  863  864  865  866  867  868  869  870  871
##  [872]  872  873  874  875  876  877  878  879  880  881  882  883  884
##  [885]  885  886  887  888  889  890  891  892  893  894  895  896  897
##  [898]  898  899  900  901  902  903  904  905  906  907  908  909  910
##  [911]  911  912  913  914  915  916  917  918  919  920  921  922  923
##  [924]  924  925  926  927  928  929  930  931  932  933  934  935  936
##  [937]  937  938  939  940  941  942  943  944  945  946  947  948  949
##  [950]  950  951  952  953  954  955  956  957  958  959  960  961  962
##  [963]  963  964  965  966  967  968  969  970  971  972  973  974  975
##  [976]  976  977  978  979  980  981  982  983  984  985  986  987  988
##  [989]  989  990  991  992  993  994  995  996  997  998  999 1000 1001
## [1002] 1002 1003 1004 1005 1006 1007 1008 1009 1010 1011 1012 1013 1014
## [1015] 1015 1016 1017 1018 1019 1020 1021 1022 1023 1024 1025 1026 1027
## [1028] 1028 1029 1030 1031 1032 1033 1034 1035 1036 1037 1038 1039 1040
## [1041] 1041 1042 1043 1044 1045 1046 1047 1048 1049 1050 1051 1052 1053
## [1054] 1054 1055 1056 1057 1058 1059 1060 1061 1062 1063 1064 1065 1066
## [1067] 1067 1068 1069 1070 1071 1072 1073 1074 1075 1076 1077 1078 1079
## [1080] 1080 1081 1082 1083 1084 1085 1086 1087 1088 1089 1090 1091 1092
## [1093] 1093 1094 1095 1096 1097 1098 1099 1100 1101 1102 1103 1104 1105
## [1106] 1106 1107 1108 1109 1110 1111 1112 1113 1114 1115 1116 1117 1118
## [1119] 1119 1120 1121 1122 1123 1124 1125 1126 1127 1128 1129 1130 1131
## [1132] 1132 1133 1134 1135 1136 1137 1138 1139 1140 1141 1142 1143 1144
## [1145] 1145 1146 1147 1148 1149 1150 1151 1152 1153 1154 1155 1156 1157
## [1158] 1158 1159 1160 1161 1162 1163 1164 1165 1166 1167 1168 1169 1170
## [1171] 1171 1172 1173 1174 1175 1176 1177 1178 1179 1180 1181 1182 1183
## [1184] 1184 1185 1186 1187 1188 1189 1190 1191 1192 1193 1194 1195 1196
## [1197] 1197 1198 1199 1200 1201 1202 1203 1204 1205 1206 1207 1208 1209
## [1210] 1210 1211 1212 1213 1214 1215 1216 1217 1218 1219 1220 1221 1222
## [1223] 1223 1224 1225 1226 1227 1228 1229 1230 1231 1232 1233 1234 1235
## [1236] 1236 1237 1238 1239 1240 1241 1242 1243 1244 1245 1246 1247 1248
## [1249] 1249 1250 1251 1252 1253 1254 1255 1256 1257 1258 1259 1260 1261
## [1262] 1262 1263 1264 1265 1266 1267 1268 1269 1270 1271 1272 1273 1274
## [1275] 1275 1276 1277 1278 1279 1280 1281 1282 1283 1284 1285 1286 1287
## [1288] 1288 1289 1290 1291 1292 1293 1294 1295 1296 1297 1298 1299 1300
## [1301] 1301 1302 1303 1304 1305 1306 1307 1308 1309 1310 1311 1312 1313
## [1314] 1314 1315 1316 1317 1318 1319 1320 1321 1322 1323 1324 1325 1326
## [1327] 1327 1328 1329 1330 1331 1332 1333 1334 1335 1336 1337 1338 1339
## [1340] 1340 1341 1342 1343 1344 1345 1346 1347 1348 1349 1350 1351 1352
## [1353] 1353 1354 1355 1356 1357 1358 1359 1360 1361 1362 1363 1364 1365
## [1366] 1366 1367 1368 1369 1370 1371 1372 1373 1374 1375 1376 1377 1378
## [1379] 1379 1380 1381 1382 1383 1384 1385 1386 1387 1388 1389 1390 1391
## [1392] 1392 1393 1394 1395 1396 1397 1398 1399 1400 1401 1402 1403 1404
## [1405] 1405 1406 1407 1408 1409 1410 1411 1412 1413 1414 1415 1416 1417
## [1418] 1418 1419 1420 1421 1422 1423 1424 1425 1426 1427 1428 1429 1430
## [1431] 1431 1432 1433 1434 1435 1436 1437 1438 1439 1440 1441 1442 1443
## [1444] 1444 1445 1446 1447 1448 1449 1450 1451 1452 1453 1454 1455 1456
## [1457] 1457 1458 1459 1460 1461 1462 1463 1464 1465 1466 1467 1468 1469
## [1470] 1470 1471 1472 1473 1474 1475 1476 1477 1478 1479 1480 1481 1482
## [1483] 1483 1484 1485 1486 1487 1488 1489 1490 1491 1492 1493 1494 1495
## [1496] 1496 1497 1498 1499 1500 1501 1502 1503 1504 1505 1506 1507 1508
## [1509] 1509 1510 1511 1512 1513 1514 1515 1516 1517 1518 1519 1520 1521
## [1522] 1522 1523 1524 1525 1526 1527 1528 1529 1530 1531 1532 1533 1534
## [1535] 1535 1536 1537 1538 1539 1540 1541 1542 1543 1544 1545 1546 1547
## [1548] 1548 1549 1550 1551 1552 1553 1554 1555 1556 1557 1558 1559 1560
## [1561] 1561 1562 1563 1564 1565 1566 1567 1568 1569 1570 1571 1572 1573
## [1574] 1574 1575 1576 1577 1578 1579 1580 1581 1582 1583 1584 1585 1586
## [1587] 1587 1588 1589 1590 1591 1592 1593 1594 1595 1596 1597 1598 1599
## [1600] 1600 1601 1602 1603 1604 1605 1606
```

R list object

1. Calculate the `summary()` of `m1` and save it as `m2` .

```
m1 <- lm(O2 ~ Depth, data=dat)
m2 <- summary(m1)
m2
```

```
## 
## Call:
## lm(formula = O2 ~ Depth, data = dat)
## 
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -173.900  -32.462   -2.995   31.631  164.641 
## 
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 190.52676    2.63921   72.19   <2e-16 ***
## Depth        -1.18384    0.02274  -52.06   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 43.25 on 1247 degrees of freedom
##   (81 observations deleted due to missingness)
## Multiple R-squared:  0.6849, Adjusted R-squared:  0.6847 
## F-statistic:  2711 on 1 and 1247 DF,  p-value: < 2.2e-16
```

2. What is the class and mode of `m2`?

```
class(m2)
```

```
## [1] "summary.lm"
```

```
mode(m2)
```

```
## [1] "list"
```

3. Using a single line of code, pull out just the p-values from `m2`.
    - *Hint*: You will need to use both `$` and `[ ]`.

```
m2$coefficients[,"Pr(>|t|)"]
```

```
##   (Intercept)         Depth 
##  0.000000e+00 5.078467e-315 
```

S4 objects

1. Compute and store the variance-covariance matrix of `m3` using `vcov()`.

```
m3 <- lmer(O2 ~ Cruise + (0 + Cruise | Depth), dat)
vcov.m3 <- vcov(m3)
```

2. What class and mode is it?

```
class(vcov.m3)
```

```
## [1] "dpoMatrix"
## attr(,"package")
## [1] "Matrix"
```

```
mode(vcov.m3)
```

```
## [1] "S4"
```

3. What elements does it contain?

```
attributes(vcov.m3)
```

```
## $x
## [1] 11.81234974 -0.19628524 -0.19628524  0.05874587
## 
## $Dim
## [1] 2 2
## 
## $Dimnames
## $Dimnames[[1]]
## [1] "(Intercept)" "Cruise"
## 
## $Dimnames[[2]]
## [1] "(Intercept)" "Cruise"
## 
## 
## $uplo
## [1] "U"
## 
## $factors
## $factors$correlation
## 2 x 2 Matrix of class "corMatrix"
##             (Intercept)     Cruise
## (Intercept)   1.0000000 -0.2356301
## Cruise       -0.2356301  1.0000000
## 
## 
## $class
## [1] "dpoMatrix"
## attr(,"package")
## [1] "Matrix"
```

4. What are the dimensions of `factors` within this object?

```
dim(vcov.m3@factors$correlation)
```

```
## [1] 2 2
```

Functions

Basics

1. Put the following math into a function

$$f(x) = 1 + 2x - 5x^2 + x^3$$

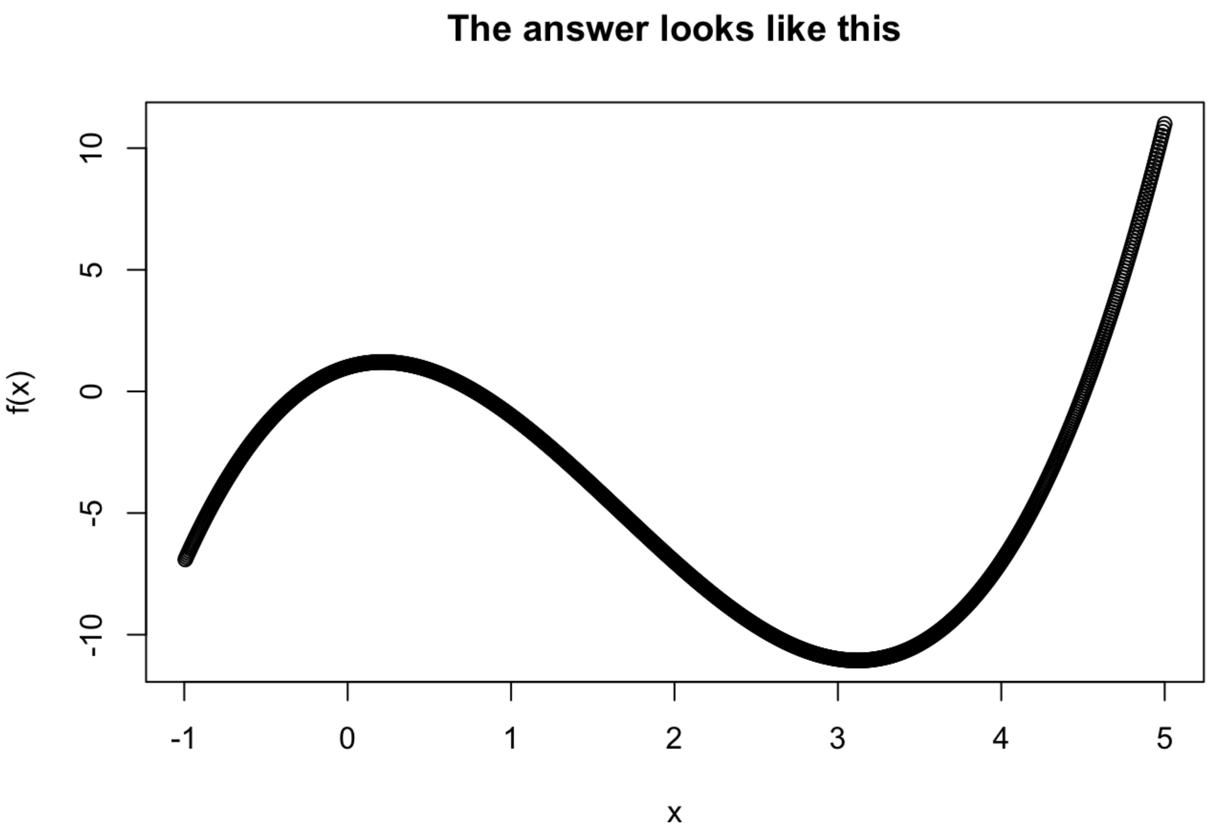
```
f <- function(x) {
  1 + 2*x - 5*x^2 + x^3
}
```

2. Set x to `1:1000/1000*6-1`

```
x <- 1:1000/1000*6-1
```

3. Plot the results with

```
plot(x, f(x) , main="The answer looks like this")
```



Scoping

1. Remove all instances of x, z, and f( ) from your environment so that you are starting fresh for this exercise.

```
rm(x)
rm(z)
```

```
## Warning in rm(z): object 'z' not found
```

```
rm(f)
```

2. What happens when we run `f()` ? Why?
    - x is not defined anywhere

```
f <- function()
{
  return(2*x)
}

f()
```

3. What will `f()` return? Why?
    - The definition of x within the function (2) overrides the global definition of x (1)

```
x <- 1

f <- function()
{
  x = 2
  return(2*x)
}

f()
```

4. What does the final `y` call return?
    - The defintion of y within the function does not alter the global definition of y outside of the function

```
y <- 1

f <- function(x)
{
  y = x+2
  return(x*y)
}

f(1)

y
```

Building a function

Steps 1-5

```
lm.function <- function(data, cruise, x, y){
  # Load necessary packages
  require(tidyverse)

  # Subset the data to the cruise of interest
  dat.subset <- data %>% filter(Cruise == cruise)

  for(y.variable in y){ # Loop through all variables provided in y
    # Fit a linear model
    model <- lm(dat.subset[[y.variable]] ~ dat.subset[[x]])
    # Summarize the model
    sum <- summary(model)
    # Extract p-values from the summary
    pval <- sum$coefficients[,"Pr(>|t|)"]

    # Print p-values to the console
    print(pval)
  }
}
```

1. Apply the current `lm.function` to all the available geochemical variables in the Saanich data set. Which ones appear to be significantly correlated with depth?

```
lm.function(data=dat, cruise=72, x="Depth", y=c("Temperature","Salinity","Density","O2","NO3","H2S"))
```

```
##     (Intercept) dat.subset[[x]]
##    2.243349e-11    3.930210e-02
##     (Intercept) dat.subset[[x]]
##    6.303475e-22    1.618484e-05
##     (Intercept) dat.subset[[x]]
##    9.333568e-20    1.151126e-04
##     (Intercept) dat.subset[[x]]
##    3.636103e-07    1.079146e-05
##     (Intercept) dat.subset[[x]]
##    0.0003919827    0.0740374332
##     (Intercept) dat.subset[[x]]
##     0.055483631     0.003832347
```

2. Copy the `lm.function` and alter it to print out the models' adjusted R-squared values instead of p-values. Be sure to run the function with inputs to make sure it works!

```
lm.function.r <- function(data, cruise, x, y){
  # Load necessary packages
  require(tidyverse)

  # Subset the data to the cruise of interest
  dat.subset <- data %>% filter(Cruise == cruise)

  for(y.variable in y){ # Loop through all variables provided in y
    # Fit a linear model
    model <- lm(dat.subset[[y.variable]] ~ dat.subset[[x]])
    # Summarize the model
    sum <- summary(model)
    # Extract p-values from the summary
    r <- sum$adj.r.squared

    # Print p-values to the console
    print(r)
  }
}

lm.function.r(data=dat, cruise=72, x="Depth", y=c("O2","NO3"))
```

```
## [1] 0.7429309
## [1] 0.1538483
```

Steps 6-7

1. Using our final `lm.function`, determine the linear fits for all geochemical variables for Cruise 12.

```
lm.function <- function(data, cruise, x, y){
  # Load necessary packages
  require(tidyverse)
    # Remove old results file, if exists
  if(file.exists("pval_results.csv")){file.remove("pval_results.csv")}

  # Subset the data to the cruise of interest
  dat.subset <- data %>% filter(Cruise == cruise)

  for(y.variable in y){ # Loop through all variables provided in y
    # Fit a linear model
    model <- lm(dat.subset[[y.variable]] ~ dat.subset[[x]])
    # Summarize the model
    sum <- summary(model)
    # Extract p-values from the summary
    # Reformat to a 1x2 data frame
    pval <- as.data.frame(t(sum$coefficients[,"Pr(>|t|)"]))
    # Add y variable name label
    pval$variable <- y.variable

    # Print p-values to a table
    write_csv(pval, path="pval_results.csv", append=TRUE)
  }
  # Create dynamic names for columns
  col1 <- paste(colnames(pval)[1], "p", sep=".")
  col2 <- paste(x, "p", sep=".")

  # Create dynamic name fo results table
  table.name <- paste(x, "lm_pvals.csv", sep="_")

  # Read in p-value results and add column names
  read_csv("pval_results.csv", col_names=FALSE) %>%
    rename(!!as.name(col1) := X1,
           !!as.name(col2) := X2,
                   variable = X3) %>%
  # Re-write the results, now with column names
  write_csv(path=table.name)
}
```

```
lm.function(data=dat, cruise=72, x="Depth", y=c("Temperature","Salinity","Densi
ty","O2","NO3","H2S"))
```

```
## Parsed with column specification:
## cols(
##   X1 = col_double(),
##   X2 = col_double(),
##   X3 = col_character()
## )
```

```
read_csv("Depth_lm_pvals.csv")
```

```
## Parsed with column specification:
## cols(
##   `(Intercept).p` = col_double(),
##   Depth.p = col_double(),
##   variable = col_character()
## )
```

(Intercept).p	Depth.p	variable
<dbl>	<dbl>	<chr>
2.243349e-11	3.930210e-02	Temperature
6.303475e-22	1.618484e-05	Salinity
9.333568e-20	1.151126e-04	Density
3.636103e-07	1.079146e-05	O2
3.919827e-04	7.403743e-02	NO3
5.548363e-02	3.832347e-03	H2S

6 rows

2. Choose a different x variable and determine if any of the Saanich geochemical variables correlate with it.

```
lm.function(data=dat, cruise=72, x="Temperature", y=c("Depth","Salinity","Density","O2","NO3","H2S"))
```

```
## Parsed with column specification:
## cols(
##   X1 = col_double(),
##   X2 = col_double(),
##   X3 = col_character()
## )
```

```
read_csv("Temperature_lm_pvals.csv")
```

```
## Parsed with column specification:
## cols(
##   `(Intercept).p` = col_double(),
##   Temperature.p = col_double(),
##   variable = col_character()
## )
```

(Intercept).p	Temperature.p	variable
<dbl>	<dbl>	<chr>
5.780941e-03	3.930210e-02	Depth
7.887599e-15	3.569910e-05	Salinity

(Intercept).p	Temperature.p	variable
<dbl>	<dbl>	<chr>
4.562811e-15	2.112427e-06	Density
4.617116e-04	1.256447e-04	O2
1.401023e-02	5.838638e-02	NO3
7.284870e-01	8.665078e-01	H2S

6 rows