Accessing BacDive API using R

Anna Vetcininova, Johannes Sikorski, Lorenz Reimer, others
DSMZ, Braunschweig

October 12, 2016

Abstract

This file documents some examples of how to access the BacDive API using R. This analysis is shown in the file "BacDive.pdf" in folder "E://Eigene Dateien//Korrespondenz//Lorenz////knitr". The R code is embedded.

Contents

List of Tables

List of Figures

This is a `knitr` file. `knitr` is a tool that allows to embed the R code for complete data analyses in LaTeX documents. The purpose is to create dynamic reports, which can be updated automatically if data or analysis change. Instead of inserting a prefabricated graph or table into the report, the master document contains the R code necessary to obtain it. When run through R, all data analysis output (tables, graphs, etc.) is created on the fly and inserted into a final LaTeX document. The report can be automatically updated if data or analysis change, which allows for truly reproducible research.

For the purpose of understanding how the respective statistics and graphics have been made, the echoing of the R commands is not supressed.

Load the necessary libraries.

```
rm(list=ls(all=TRUE)) # removes all functions, libraries, data etc

library(RCurl)

## Loading required package:  bitops

library(rjson)
library(ggplot2)
```

1 Comparison of optimum growth temperatures of the genera *Bacillus* and *Pseudomonas*

Strategie: we first need to identify the BacDive-IDs for all strains of *Bacillus* and *Pseudomonas*. We then use the BacDive-IDs to retrieve from the details strain informations the growth temperatures.

1.1 Retrieve *Pseudomonas* data

Extract BacDive IDs for all *Pseudomonas*. Then use the BacDive IDs to extract the temperatures.

Note: you need to enter your login and password.

```
### loop to collect all the ID numbers of the target taxon
allPseud_ID <- c() # empty vector to collect all taxon IDs
pages <- c(1:6) # how much pages (of each 100 BacDive IDs) do you expect? Check first at http://bacdive.dsmz.de/
counter <- 1 # counts through the set of pages (each page as 100 entries)

repeat{
  x <- getURL(URLencode(paste0("https://bacdive.dsmz.de/api/bacdive/taxon/Pseudomonas/",
                               "?page=", pages[counter],'&format=json')),
                    userpwd="your_login:your_password", httpauth = 1L)

  # convert to list, then extract the ID values into a numerical vector
  xx <- fromJSON(x)
  STR <- strsplit(unlist(xx$results), "/")
  allPseud_ID <-  c(allPseud_ID, as.numeric(sapply(STR,function(x) x[7])))

  if (counter == length(pages)) break
  counter <- counter + 1
}

# take the ID values, search in BacDive with that ID value, and extract the temperature value
counter <- 1
temperature_PS <- c()

repeat{
  print(counter)
  xlist <- fromJSON(getURL(URLencode(paste0('https://bacdive.dsmz.de/api/bacdive/bacdive_id/',
                                  allPseud_ID[counter],'/?format=json')),
                           userpwd="your_login:your_password", httpauth = 1L))
  xID <- as.numeric(xlist$culture_growth_condition$culture_temp[[1]]$temp)
  temperature_PS <- c(temperature_PS, xID)
  if (counter == length(allPseud_ID)) break
  counter <- counter + 1
}
length(temperature_PS)
```

Show the IDs (not ordered) and temperature values (sorted from smallest to largest). Obviously, not for each ID a temperature value is present.

```
options(width = 170)
allPseud_ID
```

```
##   [1]   329 12757 12758 12759 12760 12761 12762 12763 12764 12765 12766 12767 12768 12769 12770 12771 12772 12773 12774 12775 12776 12777 12778
##  [24] 12779 12780 12781 12782 12783 12784 12785 12786 12787 12788 12789 12790 12791 12792 12793 12794 12795 12796 12797 12798 12799 12800 12801
##  [47] 12802 12803 12804 12805 12806 12807 12808 12809 12810 12811 12812 12813 12814 12815 12816 12817 12818 12819 12820 12821 12822 12823 12824
##  [70] 12825 12826 12827 12828 12829 12830 12831 12832 12833 12834 12835 12836 12837 12838 12839 12840 12841 12842 12843 12844 12845 12846 12847
##  [93] 12848 12849 12850 12851 12852 12853 12854 12855 12856 12857 12858 12859 12860 12861 12862 12863 12864 12865 12866 12867 12868 12869 12870
## [116] 12871 12872 12873 12874 12875 12876 12877 12878 12879 12880 12881 12882 12883 12884 12885 12886 12887 12888 12889 12890 12891 12892 12893
## [139] 12894 12895 12896 12897 12898 12899 12900 12901 12902 12903 12904 12905 12906 12907 12908 12909 12910 12911 12912 12913 12914 12915 12916
## [162] 12917 12918 12919 12920 12921 12922 12923 12924 12925 12926 12927 12928 12929 12930 12931 12932 12933 12943 12944 12945 12946 12947 12948
## [185] 12949 12950 12951 12952 12953 12955 12956 12957 12958 12959 12960 12961 12962 12963 12964 12965 12966 12967 12968 12969 12970 12971 12972
## [208] 12973 12974 12975 12976 12977 12978 12979 12980 12981 12982 12983 12984 12985 12986 12987 12988 12989 12990 12991 12992 12993 12994 12995
## [231] 12996 12997 12998 12999 13000 13001 13002 13003 13004 13005 13006 13007 13008 13009 13010 13011 13012 13013 13014 13015 13016 13017 13019
## [254] 13020 13021 13022 13023 13024 13025 13026 13027 13028 13029 13030 13031 13032 13033 13034 13035 13036 13037 13038 13039 13040 13041 13042
```

```
## [277]  13043  13044  13045  13046  13047  13048  13049  13051  13052  13053  13054  13055  13056  13057  13058  13059  13060  13061  13062  13063  13064  13065  13066
## [300]  13067  13068  13069  13070  13071  13072  13073  13074  13075  13076  13077  13078  13079  13080  13081  13082  13083  13084  13085  13086  13087  13088  13089
## [323]  13090  13091  13092  13093  13094  13095  13096  13097  13098  13099  13100  13101  13102  13103  13104  13105  13106  13107  13108  13109  13110  13111  13112
## [346]  13113  13114  13115  13116  13117  13118  13119  13120  13121  13122  13123  13124  13125  13126  13127  13128  13129  13130  13131  13132  13133  13134  13135
## [369]  13136  13137  13138  13139  13140  13141  13143  13144  13145  13147  13148  13149  13150  13151  13152  13153  13154  13155  13156  13157  13158  13159  13160
## [392]  13161  13162  13163  13164  13165  13166  13167  13168  13169  13170  13171  13172  13173  13174  13175  13176  13177  13178  13179  13180  13181  13182  13183
## [415]  13184  13185  13186  13187  13188  13189  13190  23205  23206  23207  23208  23209  23210  23211  23212  23213 100159 100160 100161 100162 100163 100164 100165
## [438] 100166 100167 100168 100169 100170 100171 100172 100173 100174 100175 100176 100177 100178 100179 100180 100181 100182 100183  24353  24354  24355  24356  24357
## [461]  24358  24359  24360  24361  24368  24369  24370  24371  24372  24373  24382  24383  24384  24385  24386  24387  24388  24389  24390  24391  24392  24393  24394
## [484]  24395  24396  24397  24398  24399  24400  24401  24420  24421  24432  24433  24434  24435  24436  24437  24438  24439  24440  24441  24442  24864  24865  24866
## [507]  24867  24868  24869 130547 130548 130550 130551 130552 130553 130554 130555 130556 130557 130558 130559 130560 130561 130562 130563 130564 130565 130566 130567
## [530] 130568 130569 130570 130571 130572 130573 130574 130575 130576 130577 130578 130579 130580 130581 130995 130996 130997 130998 130999 131000 131001 131002 131003
## [553] 131005 131006 131007 131008 131009 131069 131075 131078 131085 131114 131382
```

```
length(allPseud_ID)
```

```
## [1] 563
```

```
sort(temperature_PS)
```

```
##   [1] 18 20 20 20 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26
##  [56] 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 26 27 27 27 27 27 27 27 27 27 27 27 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28
## [111] 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28
## [166] 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28
## [221] 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
## [276] 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
## [331] 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
## [386] 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37
## [441] 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 50
```

```
length(temperature_PS)
```

```
## [1] 461
```

1.2 Retrieve *Bacillus* data

Extract BacDive IDs for all *Bacillus.* Then use the BacDive IDs to extract the temperatures.

Note: you need to enter your login and password.

```
### loop to collect all the ID numbers of the target taxon
allBacillus_ID <- c() # empty vector to collect all taxon IDs
pages <- c(1:10) # how much pages (of each 100 BacDive IDs) do you expect? Check first at http://bacdive.dsmz.de/
counter <- 1 # counts through the set of pages (each page as 100 entries)

repeat{
  print(counter)
  x <- getURL(URLencode(paste0("https://bacdive.dsmz.de/api/bacdive/taxon/Bacillus/",
                               "?page=", pages[counter],'&format=json')),
                    userpwd="your_login:your_password", httpauth = 1L)

  # convert to list, then extract the ID values into a numerical vector
  xx <- fromJSON(x)
  STR <- strsplit(unlist(xx$results), "/")
  allBacillus_ID <-  c(allBacillus_ID, as.numeric(sapply(STR,function(x) x[7])))

  if (counter == length(pages)) break
  counter <- counter + 1
}

# take the ID values, search in BacDive with that ID value, and extract the temperature value
counter <- 1
temperature_Bac <- c()

repeat{
  print(counter)
  xlist <- fromJSON(getURL(URLencode(paste0('https://bacdive.dsmz.de/api/bacdive/bacdive_id/',
                                    allBacillus_ID[counter],'/?format=json')),
                           userpwd="your_login:your_password", httpauth = 1L))
  xID <- as.numeric(xlist$culture_growth_condition$culture_temp[[1]]$temp)
  temperature_Bac <- c(temperature_Bac, xID)
  if (counter == length(allBacillus_ID)) break
  counter <- counter + 1
}
length(temperature_Bac)
```

Extract BacDive IDs for all Bacillus. Then use the BacDive IDs to extract the temperatures.

Show the IDs (not ordered) and temperature values (sorted from smallest to largest). Obviously, not for each ID a temperature value is present.

```r
options(width = 170)
allBacillus_ID
```

```
##   [1]    572    573    574    575    576    577    578    579    580    581    582    583    584    585    586    587    588    589    590    591    592    593    594
##  [24]    595    596    597    598    599    600    601    602    603    604    605    606    607    608    609    610    611    612    613    614    615    616    617
##  [47]    618    620    621    622    623    624    625    626    627    628    629    630    631    632    633    634    635    636    637    638    639    640    641
##  [70]    642    643    644    645    646    647    648    649    650    651    652    653    654    655    656    657    658    659    660    661    662    663    664
##  [93]    665    666    667    668    669    670    671    672    673    674    675    676    677    678    679    680    681    682    683    684    685    686    687
## [116]    688    689    690    691    692    693    694    695    696    697    698    699    700    701    702    703    704    705    706    707    708    709    710
## [139]    711    712    713    714    715    716    717    718    719    720    721    722    723    724    725    726    727    728    729    730    731    732    733
## [162]    734    735    736    737    738    739    740    741    742    743    744    745    746    747    748    749    750    751    752    753    754    755    756
## [185]    757    758    759    760    761    762    763    764    765    766    767    768    769    770    771    772    773    774    775    776    777    778    779
## [208]    780    781    787    788    789    790    791    792    793    794    795    796    797    798    799    800    801    802    803    804    805    806    807
## [231]    808    809    810    811    812    813    814    815    816    817    818    819    820    821    822    823    824    825    826    827    828    829    830
## [254]    831    832    833    834    835    836    837    838    839    840    841    842    843    844    845    846    847    848    849    850    851    852    853
## [277]    854    855    856    857    858    859    860    861    862    863    864    865    866    867    868    869    870    871    872    873    874    875    876
## [300]    877    878    879    880    881    882    883    884    885    886    887    888    889    890    891    892    893    894    895    896    897    898    899
## [323]    900    901    902    903    904    905    906    907    908    909    910    911    912    913    914    915    916    917    918    919    920    921    922
## [346]    923    924    925    926    927    928    929    930    931    932    933    934    935    936    937    938    939    940    941    942    943    944    945
## [369]    946    947    948    949    950    951    952    953    954    955    956    957    958    959    960    961    962    963    964    965    966    967    968
## [392]    969    970    971    972    973    974    975    976    977    978    979    980    981    982    983    984    985    986    987    988    989    990    991
## [415]    992    993    994    995    996    997    998    999   1000   1001   1002   1003   1004   1005   1006   1007   1008   1009   1010   1011   1012   1013   1014
## [438]   1015   1016   1017   1018   1019   1020   1021   1022   1023   1024   1025   1026   1027   1028   1029   1030   1031   1032   1033   1034   1035   1036   1037
## [461]   1038   1039   1040   1041   1042   1043   1044   1045   1046   1047   1048   1049   1050   1051   1052   1053   1054   1055   1056   1057   1058   1059   1060
## [484]   1061   1062   1063   1064   1065   1066   1067   1068   1069   1070   1071   1072   1073   1074   1075   1076   1077   1078   1079   1080   1081   1082   1083
## [507]   1084   1085   1086   1087   1088   1089   1090   1091   1092   1093   1094   1095   1096   1097   1098   1099   1100   1101   1102   1103   1104   1105   1106
## [530]   1107   1108   1109   1110   1111   1112   1113   1114   1115   1116   1117   1118   1119   1120   1121   1122   1123   1124   1125   1126   1127   1128   1129
## [553]   1130   1131   1132   1133   1134   1135   1136   1137   1138   1139   1140   1141   1142   1143   1144   1145   1146   1147   1148   1149   1150   1151   1152
## [576]   1153   1154   1155   1156   1157   1158   1159   1160   1161   1162   1163   1164   1165   1166   1167   1168   1169   1170   1171   1172   1173   1174   1175
## [599]   1176   1177   1178   1179   1180   1181   1182   1183   1184   1185   1186   1187   1188   1189   1190   1191   1192   1193   1194   1195   1196   1197   1198
## [622]   1199   1200   1201   1202   1203   1204   1207   1208   1209   1210   1211   1212   1213   1214   1215   1216   1217   1219   1220   1221   1222   1223   1224
## [645]   1225   1226   1227   1228   1229   1230   1231   1232   1233   1234   1235   1236   1238   1239   1240   1241   1242   1243   1244   1246   1247   1248   1249
## [668]   1250   1251   1252   1253   1254   1255   1256   1257   1258   1259   1260   1261   1262   1263   1264   1265   1266   1267   1268   1269   1270   1271   1273
## [691]   1274   1275   1277   1278   1279   1280   1281   1282   1283   1284   1286   1287   1288   1289   1290   1291   1292   1293   1294   1295   1296   1297   1299
## [714]   1300   1302   1303   1305   1306   1307   1308   1309   1310   1311   1312   1314   1315   1316   1317   1318   1319   1320   1321   1322   1323 130881 130880
## [737] 130877 130878 130879 130876 130874 130875 130872 130873 130870 130871 130869 130867 130868 130865 130866 130864 130862 130863 130861 130860 130859 130857 130858
## [760] 130856  22911  22912  22913  22914  22915  22916  22917  22918  22919  22920  22921  22922  22923  22924  22925  22926  22927 100131 100132 100133 100134 100135
## [783] 100136 100137 100138 100139 100140 100141  23587  23588  23589  23590  23591  23592  23593  23594  23595  23596  23597  23598  23599  23600  23601  23602  23615
## [806]  23616  23635  23636  23637  23638  23639  23640  23641  23642  23643  23644  23645  23646  23647  23648  23649  23650  23651  23652  23653  23654  23655  23656
## [829]  23657  23658  23659  23660  23661  23662  23663  23664  23665  23666  23667  23668  23670  23671  23672  23673  23674  23675  23676  23677  23678  23679  23680
## [852]  23681  23692  23693  23694  23695  23696  23697  23698  23699  23700  23701  23702  23706  23707  23708  23709  23710  23711  23712  23713  23714  23715  23716
## [875]  23717  23718  24827  24828  24829  24830  24831  24832  24833  24834  24835  24836  24837  24838  24839  24840  24841 130161 130162 130163 130164 130165 130166
## [898] 130167 130168 130169 130170 130171 130172 130173 130174 130175 130176 130177 130178 130179 130180 130181 130182 130183 130184 130185 130186 130187 130188 130189
## [921] 130190 130191 130192 130193 130194 130195 130196 130197 130198 130199 131049 131050 131081 131177 131356 131357 131447 131455
```

```r
length(allBacillus_ID)
```

```
## [1] 938
```

```r
sort(temperature_Bac)
```

```
##   [1] 20 20 20 20 20 20 20 22 22 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 25 26 26 26 26 26 26 26 26 26 26 26 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28
##  [56] 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28
## [111] 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28
## [166] 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28
## [221] 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 28 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
## [276] 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
## [331] 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
## [386] 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
## [441] 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
## [496] 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
## [551] 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
## [606] 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
## [661] 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30
## [716] 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 30 32 32 35 35 35 35 37 37 37 37 37
## [771] 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 37 40 40 40 42 42 45 45 45 45 45 45 45 45 50 55 55 55 55 55 55 55 55 55 55 55 55 55 55 55
## [826] 55 55 55 55 55 55 55 55 55 60 60 60 60 60 70 70 70 70
```

```r
length(temperature_Bac)
```

```
## [1] 843
```

2 Analyse the data

2.1 Numerical summary statistics

```
########### summarizing statistics of Pseudomonas growth temperature
summary(temperature_PS)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   18.00   28.00   28.00   29.17   30.00   50.00

# tabulate the occurence per temperature
table(temperature_PS)

## temperature_PS
##  18  20  25  26  27  28  30  37  50
##   1   3  21  47  10 173 161  44   1

###########  summarizing statistics of Bacillus growth temperature
summary(temperature_Bac)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   20.00   28.00   30.00   30.86   30.00   70.00

# tabulate the occurence per temperature
table(temperature_Bac)

## temperature_Bac
##  20  22  25  26  28  30  32  35  37  40  42  45  50  55  60  70
##   7   2  18  11 200 521   2   4  31   3   2   8   1  24   5   4
```

2.2 Graphical analysis

Below is the code for Figure 1, page 6.

```
# prepare input data for graphical analysis
data_PS <- data.frame(temperature = temperature_PS, taxon = rep("Pseudomonas", length(temperature_PS)))
data_Bac <- data.frame(temperature = temperature_Bac, taxon = rep("Bacillus", length(temperature_Bac)))
data <- rbind(data_PS, data_Bac)
head(data)

##   temperature       taxon
## 1          30 Pseudomonas
## 2          37 Pseudomonas
## 3          30 Pseudomonas
## 4          37 Pseudomonas
## 5          37 Pseudomonas
## 6          37 Pseudomonas

pdf("PDF_OUTPUT_temperature.pdf", height = 5, width = 10)
ggplot(data, aes(temperature, fill = taxon, colour = taxon)) +
  theme_bw() +
  geom_density(alpha = 0.1) +
  theme(legend.position = "bottom")
dev.off()

## pdf
##   2
```

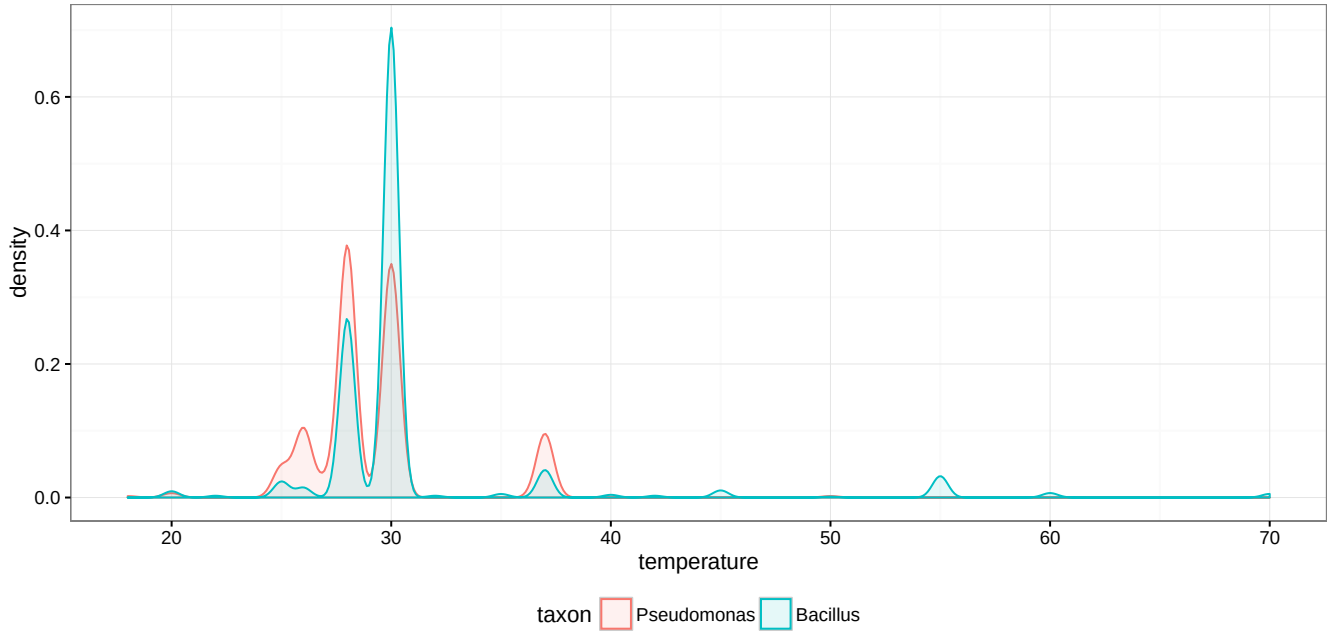

Figure 1: Density plot of optimum growth values of *Bacillus* and *Pseudomonas* strains as present in the BacDive database.

2.3 Concluding statistics using t-test

```
t.test(temperature_PS, temperature_Bac)

##
##  Welch Two Sample t-test
##
## data:  temperature_PS and temperature_Bac
## t = -6.5147, df = 1299.4, p-value = 1.039e-10
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -2.199993 -1.181660
## sample estimates:
## mean of x mean of y
##  29.16920  30.86002
```