地理建模实验5 实验报告

42109232　吕文博　地信2101班

2024-06-03

非线性回归

加载数据

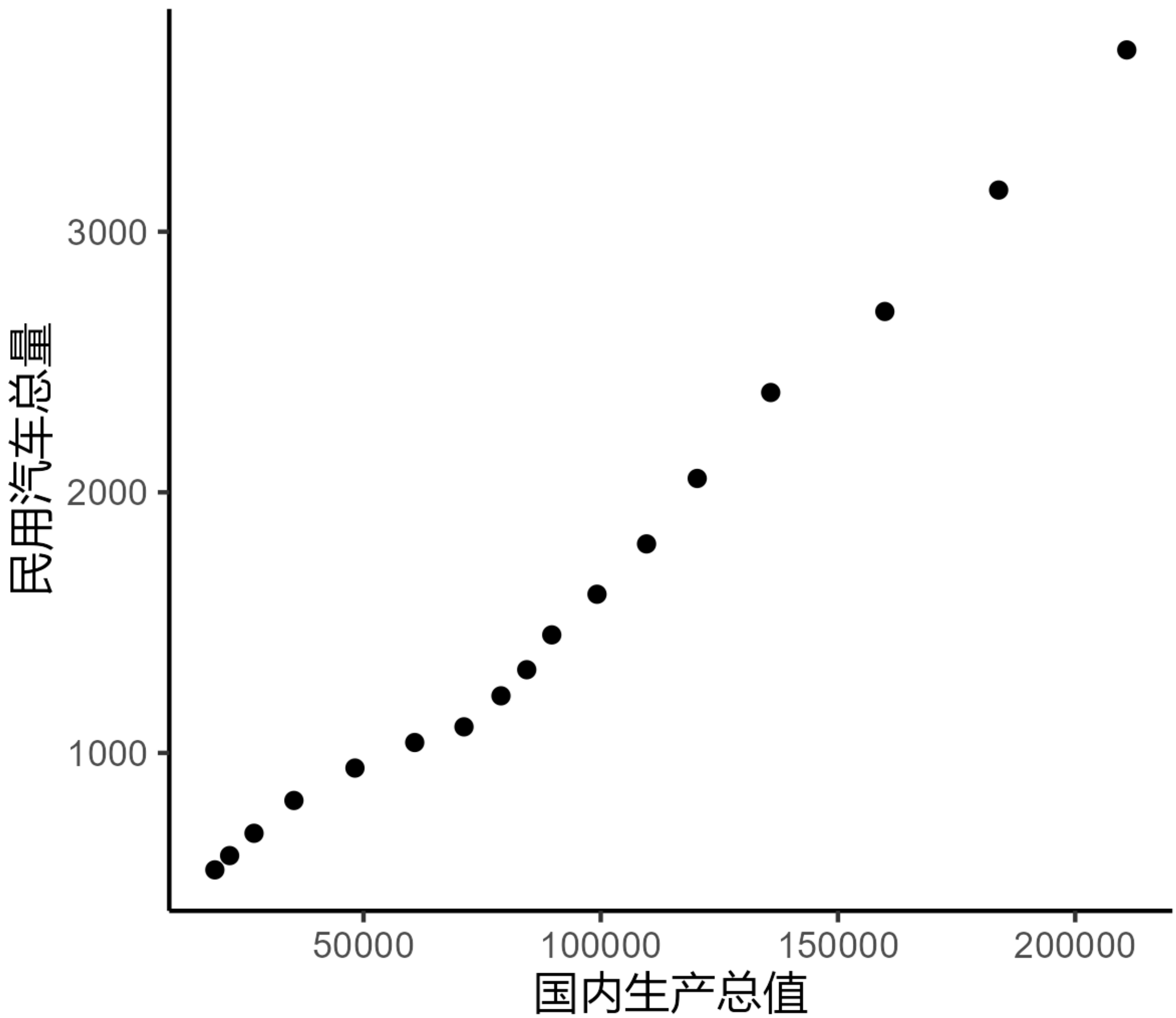
```
dt = readxl::read_xls('../data/exp5/5.xls',sheet = "非线性分析")
head(dt)
```

```
## # A tibble: 6 x 3
##    年度 国内生产总值 民用汽车总量
##   <dbl>        <dbl>        <dbl>
## 1  1990       18668.         551.
## 2  1991       21782.         606.
## 3  1992       26924.         692.
## 4  1993       35334.         818.
## 5  1994       48198.         942.
## 6  1995       60794.        1040
```

绘制散点图

```
library(ggplot2)

ggplot() +
  geom_point(data = dt,
             aes(x = 国内生产总值,
                 y = 民用汽车总量)) +
  theme_classic()
```



常见拟合曲线形式

- `linear`线性模型 $y = b_0 + b_1 x$
- `quadratic`二次模型 $y = b_0 + b_1 x + b_2 x^2$
- `compound`复合模型 $y = b_0 {b_1}^x$
- `growth`生长模型 $y = e^{(b_0 + b_1 x)}$
- `logarithmic`对数模型 $y = b_0 + b_1 \ln(x)$
- `s-curve`S形模型 $y = e^{(b_0 + \frac{b_1}{x})}$
- `cubic` 三次模型 $y = b_0 + b_1 x + b_2 x^2 + b_3 x^3$
- `exponential`指数模型 $y = b_0 e^{b_1 x}$
- `inverse`倒数模型 $y = b_0 + \frac{b_1}{x}$
- `power`幂函数模型 $y = b_0 x^{b_1}$
- `logistic`逻辑斯蒂模型 $y = \frac{1}{\frac{1}{u} + b_0 {b_1}^x}$

```r
performance_nls = \(model){
  require(minpack.lm)
  y_hat = as.double(fitted(model))
  res = as.double(residuals(model))
  y = y_hat + res
  ymean = mean(y)
  df = summary(model)$df
  df1 = length(y) - df[2] - 1
  df2 = df[2]
  SSE = sum(res^2)
  SSR = sum((y_hat-ymean)^2)
  SST = SSE + SSR
  fv = (df2 * SSR) / (df1 * SSE)
  pv = pf(fv,df1 = df1,df2 = df2,lower.tail = FALSE)
  r2 = 1 - SSE/SST
  r = sqrt(r2)
  adjr2 = 1 - ((df1 + df2) * SSE) / (df2 * SST)
  pnls = list("F-statistic" = fv, "P-value" = pv, "R2" = r2, "AdjR2" = adjr2,
              "DF1" = df1, "DF2" = df2, "model_summary" = summary(model))
  class(pnls) = 'performance_nls'
  return(pnls)
}

print.performance_nls = \(resnls,...){
  print(resnls$`model_summary`)
  cat(paste0("Multiple R-squared: ",formatC(resnls$`R2`,digits = 5,format = "g"),", \t",
             "Adjusted R-squared: ",formatC(resnls$`AdjR2`,digits = 5,format = "g")),'\n')
  cat(paste0("F-statistic: ",formatC(resnls$`F-statistic`,digits = 6,format = "g"),
             " on ",resnls$`DF1`," and ",resnls$`DF2`," DF, ",
             "p-value: ",formatC(resnls$`P-value`,digits = 3,format = "e")),'\n')
}
```

直接判断拟合曲线类型

```r
library(minpack.lm)

cubic_lm = nlsLM(民用汽车总量~b0+b1*国内生产总值+b2*国内生产总值^2+b3*国内生产总值^3,data = dt)
performance_nls(cubic_lm)
```

三次函数

```
##
## Formula: 民用汽车总量 ~ b0 + b1 * 国内生产总值 + b2 * 国内生产总值^2 +
##     b3 * 国内生产总值^3
##
## Parameters:
##      Estimate Std. Error t value Pr(>|t|)
## b0  5.264e+02  7.518e+01   7.002 9.31e-06 ***
## b1  2.391e-03  2.938e-03   0.814  0.43032
## b2  1.107e-07  3.119e-08   3.547  0.00357 **
## b3 -2.426e-13  9.351e-14  -2.594  0.02226 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 59.28 on 13 degrees of freedom
##
## Number of iterations to convergence: 5
## Achieved convergence tolerance: 1.49e-08
##
## Multiple R-squared: 0.99665,     Adjusted R-squared: 0.99588
## F-statistic:  1290.7 on 3 and 13 DF, p-value: 2.464e-16
```

```
power_lm = nlsLM(民用汽车总量~b0*国内生产总值^b1,data = dt)
performance_nls(power_lm)
```

幂函数

```
##
## Formula: 民用汽车总量 ~ b0 * 国内生产总值^b1
##
## Parameters:
##    Estimate Std. Error t value Pr(>|t|)
## b0  0.02270    0.01200   1.892    0.078 .
## b1  0.97604    0.04466  21.855  8.7e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 140.5 on 15 degrees of freedom
##
```

```
## Number of iterations to convergence: 6
## Achieved convergence tolerance: 1.49e-08
##
## Multiple R-squared: 0.98016,     Adjusted R-squared: 0.97884
## F-statistic: 741.083 on 1 and 15 DF, p-value: 3.483e-14
```

三次函数拟合的R^2为0.997，幂函数拟合的R^2为0.980,三次函数拟合优于幂函数,选择三次函数作为非线性拟合的模型,根据上面的结果可得拟合的曲线为

$$\text{民用汽车总量} = 0.05264 + 0.002391 \times \text{国内生产总值} + 1.107 \times 10^{-7} \times \text{国内生产总值}^2 - 2.426 \times 10^{-13} \times \text{国内生产总值}^3$$

拟合常见的所有曲线

```
nonlinear_fit = \(formula,data,method,...){
  require(minpack.lm)
  formula = stats::as.formula(formula)
  formula.vars = all.vars(formula)
  response = data[, formula.vars[1], drop = TRUE]
  explanatory = data[,formula.vars[2], drop = TRUE]
  data = tibble::tibble(response,explanatory)

  switch(method,
         "linear" = {
           model = nlsLM(response~b0+b1*explanatory,data,...)
         },
         "quadratic" = {
           model = nlsLM(response~b0+b1*explanatory+b2*explanatory^2,data,...)
         },
         "compound" = {
           model = nlsLM(response~b0*b1^explanatory,data,...)
         },
         "growth" = {
           model = nlsLM(response~exp(b0+b1*explanatory),data,...)
         },
         "logarithmic" = {
           model = nlsLM(response~b0+b1*log(explanatory),data,...)
         },
         "s" = {
           model = nlsLM(response~exp(b0+b1/explanatory),data,...)
```

```
        },
        "cubic" = {
          model = nlsLM(response~b0+b1*explanatory+b2*explanatory^2+b3*explanatory^3,data,...)
        },
        "exponential" = {
          model = nlsLM(response~b0*exp(b1*explanatory),data,...)
        },
        "inverse" = {
          model = nlsLM(response~b0+b1/explanatory,data,...)
        },
        "power" = {
          model = nlsLM(response~b0*explanatory^b1,data,...)
        },
        "logistic" = {
          model = nlsLM(response~1/(1/u+b0*b1^explanatory),data,...)
        })

  k = coef(model)
  p = performance_nls(model)
  res = c(method = method,
          R2 = formatC(p$`R2`,digits = 5,format = "g"),
          Pvalue = formatC(p$`P-value`,digits = 3,format = "e"),k) |>
    tibble::as_tibble_row()
  return(res)
}
```

为了曲线拟合收敛的更快,我们首先对原始数据进行了log10变换，然后拟合曲线：

```
dt = dplyr::mutate(dt,dplyr::across(2:3,~log10(.x)))
c("linear","quadratic","compound","growth","logarithmic",
  "s","cubic","exponential","inverse","power","logistic") |>
  purrr::map(~nonlinear_fit('民用汽车总量~国内生产总值',dt,.x)) |>
  purrr::list_rbind() |>
  dplyr::arrange(desc(R2)) -> resnl
resnl
```

```
## # A tibble: 11 x 8
##    method      R2      Pvalue    b0                b1         b2    b3    u
##    <chr>       <chr>   <chr>     <chr>             <chr>      <chr> <chr> <chr>
##  1 cubic       0.99342 2.004e-14 -19.292459369905  14.83478~ -3.4~ 0.27~ <NA>
##  2 quadratic   0.99265 1.158e-15 10.6971682649678  -3.96254~ 0.49~ <NA>  <NA>
```

```
##  3 logistic    0.99116 4.216e-15 -0.0173352368789323 1.606621~ <NA>  <NA>  2.01~
##  4 compound    0.96916 9.569e-13 0.923491212279344   1.284891~ <NA>  <NA>  <NA>
##  5 growth      0.96916 9.569e-13 -0.0795941290068277 0.250674~ <NA>  <NA>  <NA>
##  6 exponential 0.96916 9.569e-13 0.923491221380851   0.250674~ <NA>  <NA>  <NA>
##  7 power       0.95862 8.720e-12 0.472563195180918   1.195906~ <NA>  <NA>  <NA>
##  8 linear      0.95581 1.430e-11 -0.568751829295098  0.761507~ <NA>  <NA>  <NA>
##  9 s           0.94639 6.119e-11 2.31177187541202    -5.67701~ <NA>  <NA>  <NA>
## 10 logarithmic 0.94318 9.478e-11 -2.57095454210659   3.612000~ <NA>  <NA>  <NA>
## 11 inverse     0.92902 5.061e-10 6.6557633002356     -17.0519~ <NA>  <NA>  <NA>
```

由上述结果可知，三次曲线拟合效果最好（计算差异来自数据预先统一进行了log10变换）

趋势面回归分析

加载数据

```
dt = readxl::read_xls('../data/exp5/5.xls',sheet = "趋势面分析")
head(dt)
```

```
## # A tibble: 6 x 3
##       z     x     y
##   <dbl> <dbl> <dbl>
## 1  27.6  0      1
## 2  38.4  1.1    0.6
## 3  24    1.8    0
## 4  24.7  2.95   0
## 5  32    3.4    0.2
## 6  55.5  1.8    1.7
```

定义趋势面分析函数

```
trend.surf = \(formula,data, np = 3){
  require(magrittr)
  formula = stats::as.formula(formula)
  formula.vars = all.vars(formula)
  z = data[, formula.vars[1], drop = TRUE]
  x = data[, formula.vars[2], drop = TRUE]
  y = data[, formula.vars[3], drop = TRUE]
  generatesurf = \(n,x = 'x',y = 'y'){if(n==1){return(paste0(x,'+',y))}
                    else{return(paste0(generatesurf(n-1,x,y),'+',
                                paste0(x,'^',n:0,'*',y,'^',0:n,
```

```r
                                          collapse = '+')))}}
  if(np <= 0){
    stop('Please provide a positive integar!')
    }else {
    newdata = generatesurf(np) %>%
    stringr::str_split('\\+') %>%
    purrr::pluck(1) %>%
    purrr::map_dfc(\(i) eval(parse(text = i))) %>%
    purrr::set_names(paste0('x',1:ncol(.))) %>%
    dplyr::mutate(z = z)
    }
  surf = lm('z ~ .',data = newdata)
  return(surf)
}
```

二次趋势面分析

```r
trend2 = trend.surf(z~x+y,dt,2)
summary(trend2)
```

```
##
## Call:
## lm(formula = "z ~ .", data = newdata)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.9177 -1.5751  0.3012  2.3265  8.2646
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   5.9980    10.0236   0.598  0.57146
## x1           17.4382     6.8157   2.559  0.04299 *
## x2           29.7874     9.1328   3.262  0.01721 *
## x3           -3.5883     1.4881  -2.411  0.05248 .
## x4            0.3569     1.6101   0.222  0.83192
## x5           -8.0695     2.0844  -3.871  0.00825 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 5.613 on 6 degrees of freedom
```

```
## Multiple R-squared:  0.8386, Adjusted R-squared:  0.7041
## F-statistic: 6.236 on 5 and 6 DF,  p-value: 0.02274
```

三次趋势面分析

```r
trend3 = trend.surf(z~x+y,dt,3)
summary(trend3)
```

```
##
## Call:
## lm(formula = "z ~ .", data = newdata)
##
## Residuals:
##        1        2        3        4        5        6        7        8
## -0.76502  0.05899  2.12692 -4.19806  2.98206 -0.01453 -1.17323  1.63951
##        9       10       11       12
## -0.23200 -1.20868  1.79878 -1.01474
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  -48.810     26.922  -1.813   0.2115
## x1            37.557     22.633   1.659   0.2389
## x2           130.130     43.036   3.024   0.0942 .
## x3             8.389     10.752   0.780   0.5169
## x4           -33.166     17.636  -1.881   0.2008
## x5           -62.740     22.299  -2.814   0.1065
## x6            -4.133      2.230  -1.853   0.2050
## x7             6.138      2.767   2.218   0.1568
## x8             2.566      2.991   0.858   0.4813
## x9             9.785      3.905   2.506   0.1291
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.554 on 2 degrees of freedom
## Multiple R-squared:  0.9646, Adjusted R-squared:  0.8052
## F-statistic: 6.054 on 9 and 2 DF,  p-value: 0.1498
```

①二次趋势面R^2为0.8386，三次趋势面R^2为0.9646，数据与趋势面的拟合程度比较好，三次趋势面的拟合程度更好。②二次趋势面P值为0.02274<0.05，F检验通过，说明二次趋势面显著;三次趋势面P值为0.1498>0.05， F检验不通过，说明三次趋势面不显著，选用二次趋势面进行拟合比较合理。③综上，选用二次趋势面进行拟合比较合理。