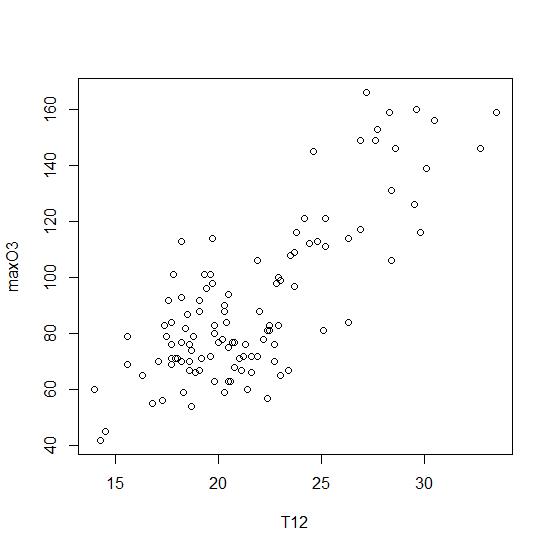
**Exemple de régression linéaire simple et multiple, régression logarithmique et exponentielle, régression polynomiale**

**Régression linéaire simple**

Dans cette partie nous allons utiliser la base de données ozone que vous pouvez importer à partir du lien suivant : <https://r-stat-sc-donnees.github.io/ozone.txt>

On peut représenter graphiquement le nuage de points maxO3 en fonction de T12 :

> plot(maxO3~T12, data=ozone)



Ce nuage de points nous fait penser à un alignement selon une forme qui n'est pas très loin d'une droite.

> reg\_simp <- lm(maxO3~T12, data=ozone)

> reg\_simp

Call:

lm(formula = maxO3 ~ T12, data = ozone)

Coefficients:

(Intercept) T12

-27.420 5.469

>

Pour plus de détail :

> summary(reg\_simp)

Call:

lm(formula = maxO3 ~ T12, data = ozone)

Residuals:

Min 1Q Median 3Q Max

-38.079 -12.735 0.257 11.003 44.671

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) -27.4196 9.0335 -3.035 0.003 \*\*

T12 5.4687 0.4125 13.258 <2e-16 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 17.57 on 110 degrees of freedom

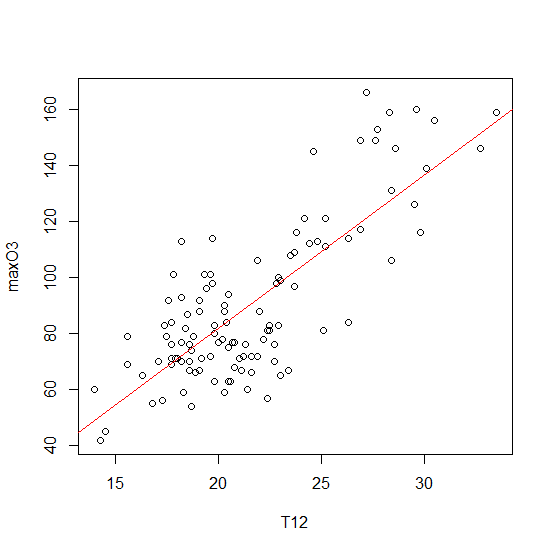
Multiple R-squared: 0.6151, Adjusted R-squared: 0.6116

F-statistic: 175.8 on 1 and 110 DF, p-value: < 2.2e-16

>

Pour tracer la droite de régression linéaire :

> abline(reg\_simp , col ="red")



Selon ce modèle de régression linéaire, prévoyons la concentration en ozone d'une journée. Sachant que la température prévue de cette journée est de T12 = 19 °C :

> a\_prevoir <- **data.frame**(T12=19)

> maxO3\_prev <- predict(reg\_simp,a\_prevoir)

> round(maxO3\_prev, digits=2)

1

76.49

>

**Régression linéaire multiple**

Nous allons utiliser la même base de données ozone. Dans cette partie nous allons chercher à expliquer maxO3 en fonction des autres variables quantitatives. En utilisant la fonction lm(), nous allons chercher le modèle de régression multiple de maxO3 en fonction des autres variable qualitatives.

> reg\_multi <- lm(maxO3~T9+T12+T15+Ne9+Ne12+Ne15+maxO3v, data=ozone)

> summary(reg\_multi)

Call:

lm(formula = maxO3 ~ T9 + T12 + T15 + Ne9 + Ne12 + Ne15 + maxO3v,

data = ozone)

Residuals:

Min 1Q Median 3Q Max

-57.768 -7.845 -1.359 8.134 38.984

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 12.70548 13.10860 0.969 0.33467

T9 -0.63596 1.03462 -0.615 0.54011

T12 2.50600 1.39946 1.791 0.07625 .

T15 0.71381 1.13674 0.628 0.53142

Ne9 -2.76057 0.89157 -3.096 0.00252 \*\*

Ne12 -0.37193 1.34590 -0.276 0.78283

Ne15 0.09028 0.99934 0.090 0.92819

maxO3v 0.37774 0.06121 6.171 1.32e-08 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 14.43 on 104 degrees of freedom

Multiple R-squared: 0.7546, Adjusted R-squared: 0.738

F-statistic: 45.68 on 7 and 104 DF, p-value: < 2.2e-16

>

On constate ici que certains paramètres ne sont pas significativement différents de 0, car leur p-valeur (la valeur indiquée par : Pr(>|t|) ) n'est pas inférieure à 5 %, le niveau de test que nous souhaitons.

Le R2 (Multiple R-squared) vaut environ 0.75, et le R2 ajusté est d'environ 0.74.

Cette valeur est plus élevée qu'en régression linéaire simple, et c'est logique, car lorsque l'on rajoute des variables explicatives potentielles, on accroît naturellement la valeur de ces R2.

**Retirez les variables non significatives**

On va maintenant retirer les variables non significatives. On commence par la moins significative, c’est-à-dire la variable qui a la p-valeur la plus grande. Dans notre exemple c’est la variable Ne15, car elle a une p-valeur de presque 0.93.

> reg\_multi = lm(maxO3~T9+T12+T15+Ne9+Ne12+maxO3v,data=ozone)#Ne15 est retirée du model

> summary(reg\_multi)

Call:

lm(formula = maxO3 ~ T9 + T12 + T15 + Ne9 + Ne12 + maxO3v, data = ozone)

Residuals:

Min 1Q Median 3Q Max

-57.689 -7.796 -1.447 8.147 38.929

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 12.84916 12.95015 0.992 0.32338

T9 -0.62976 1.02745 -0.613 0.54124

T12 2.56018 1.25841 2.034 0.04443 \*

T15 0.65787 0.94878 0.693 0.48960

Ne9 -2.76526 0.88585 -3.122 0.00232 \*\*

Ne12 -0.30796 1.13912 -0.270 0.78742

maxO3v 0.37752 0.06087 6.202 1.12e-08 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 14.36 on 105 degrees of freedom

Multiple R-squared: 0.7545, Adjusted R-squared: 0.7405

F-statistic: 53.8 on 6 and 105 DF, p-value: < 2.2e-16

>

On voit maintenant que Ne12 est la moins significative (avec une p-valeur de 0.79). On l'enlève donc.

> reg\_multi <- lm(maxO3~T9+T12+T15+Ne9+maxO3v,data=ozone)

> summary(reg\_multi)

Call:

lm(formula = maxO3 ~ T9 + T12 + T15 + Ne9 + maxO3v, data = ozone)

Residuals:

Min 1Q Median 3Q Max

-57.246 -7.607 -1.295 8.285 38.477

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 11.28440 11.53397 0.978 0.3301

T9 -0.73127 0.95218 -0.768 0.4442

T12 2.66487 1.19211 2.235 0.0275 \*

T15 0.66817 0.94386 0.708 0.4806

Ne9 -2.92578 0.65451 -4.470 1.97e-05 \*\*\*

maxO3v 0.37960 0.06012 6.314 6.48e-09 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 14.3 on 106 degrees of freedom

Multiple R-squared: 0.7544, Adjusted R-squared: 0.7428

F-statistic: 65.11 on 5 and 106 DF, p-value: < 2.2e-16

>

On constate maintenant qu’il faut retirer la variable  T9.

> reg\_multi <- lm(maxO3~T12+T15+Ne9+maxO3v,data=ozone)

> summary(reg\_multi)

Call:

lm(formula = maxO3 ~ T12 + T15 + Ne9 + maxO3v, data = ozone)

Residuals:

Min 1Q Median 3Q Max

-56.068 -7.767 -1.605 8.446 40.187

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 9.13680 11.16838 0.818 0.4151

T12 2.23175 1.04826 2.129 0.0355 \*

T15 0.62772 0.94058 0.667 0.5060

Ne9 -2.96393 0.65137 -4.550 1.42e-05 \*\*\*

maxO3v 0.37019 0.05875 6.301 6.71e-09 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 14.27 on 107 degrees of freedom

Multiple R-squared: 0.753, Adjusted R-squared: 0.7438

F-statistic: 81.55 on 4 and 107 DF, p-value: < 2.2e-16

On retire ensuite  T15

> reg\_multi <- lm(maxO3~T12+Ne9+maxO3v,data=ozone)

> summary(reg\_multi)

Call:

lm(formula = maxO3 ~ T12 + Ne9 + maxO3v, data = ozone)

Residuals:

Min 1Q Median 3Q Max

-56.385 -7.872 -1.941 7.899 41.513

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 9.76225 11.10038 0.879 0.381

T12 2.85308 0.48052 5.937 3.57e-08 \*\*\*

Ne9 -3.02423 0.64342 -4.700 7.71e-06 \*\*\*

maxO3v 0.37571 0.05801 6.477 2.85e-09 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 14.23 on 108 degrees of freedom

Multiple R-squared: 0.752, Adjusted R-squared: 0.7451

F-statistic: 109.1 on 3 and 108 DF, p-value: < 2.2e-16

>

On remarque qu’à présent, tous les paramètres sont significatifs. Quant au R2, il vaut environ 0.75, tout comme le R2 ajusté. On peut donc utiliser ce modèle à des fins de prévision.

Si l'on souhaite prévoir la concentration journalière en ozone, sachant que la température prévue à 12 h sera de 15 °C, que la valeur de Ne9 sera de 2, et que la concentration maxO3v de la veille vaut 100, alors on saisit les lignes suivantes :

> a\_prevoir <- data.frame(**T12**=15,**Ne9**=2,**maxO3v**=100)

> a\_prevoir

T12 Ne9 maxO3v

1 15 2 100

> maxO3\_prev <- **predict**(**reg\_multi**,**a\_prevoir**)

> maxO3\_prev

1

84.08126

> round(maxO3\_prev, digits=2)

1

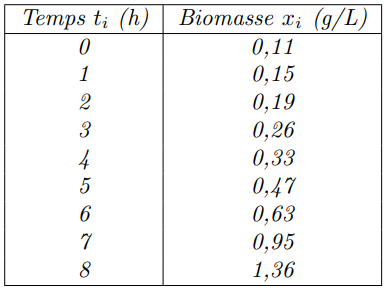
84.08

>

On obtient une concentration maxO3 de 84.

**Régression exponentiel**

L’objectif d’un ”procédé en batch” de génie fermentaire est de déterminer les caractéristiques cinétiques d’un microorganisme en particulier son taux de croissance µ. Pour cela, on va ensemencer un bioréacteur avec (entre autre) une certaine concentration de microorganismes et de substrat dont on va mesurer l’évolution au cours du temps. Des mesures de densité optique seront régulièrement effectuées à l’aide d’un spectrophotomètre. Les échantillons prélevés seront dilués afin de rester dans la zone de linéarité du spectrophotomètre, zone pour laquelle la densité optique est proportionnelle à la concentration. Après différentes calibrations, on a obtenu les concentrations suivantes :



Nous allons d’abord créer une base de données data comportant les données à traiter :

> temps = 0:8

> biomasse = c(0.11 , 0.15 , 0.19 , 0.26 , 0.33 , 0.47 , 0.63 , 0.95 , 1.36)

> data = data.frame(temps, biomasse)

> data

temps biomasse

1 0 0.11

2 1 0.15

3 2 0.19

4 3 0.26

5 4 0.33

6 5 0.47

7 6 0.63

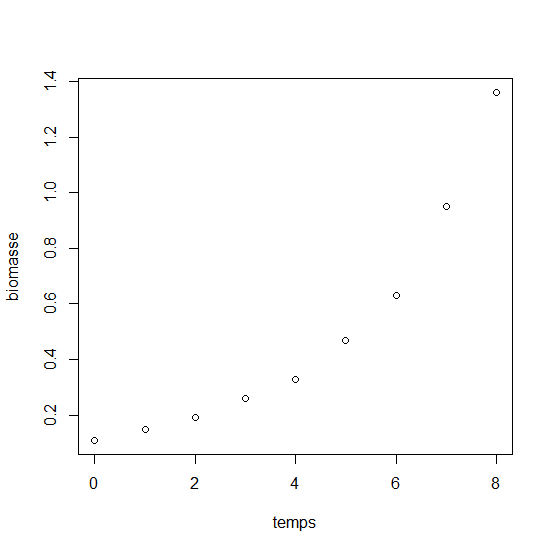
8 7 0.95

9 8 1.36

>

Ensuite nous allons afficher le nuage de points de biomasse de fonction du temps

> plot(biomasse ~ temps, data=data)



Il est clair qu’il s’agit d’une régression exponentielle. Pour trouver le model de cette régression nous allons appliquer la fonction lm() à log(biomasse) et temps :

> reg\_exp = lm (log(biomasse) ~ temps, data = data)

> reg\_exp

Call:

lm(formula = log(biomasse) ~ temps, data = data)

Coefficients:

(Intercept) temps

-2.2593 0.3098

Pour plus de détail :

> summary(reg\_exp)

Call:

lm(formula = log(biomasse) ~ temps, data = data)

Residuals:

Min 1Q Median 3Q Max

-0.08847 -0.04460 -0.01712 0.05198 0.08861

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) -2.259256 0.039685 -56.93 1.35e-10 \*\*\*

temps 0.309766 0.008336 37.16 2.66e-09 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.06457 on 7 degrees of freedom

Multiple R-squared: 0.995, Adjusted R-squared: 0.9942

F-statistic: 1381 on 1 and 7 DF, p-value: 2.656e-09

Selon ce model :

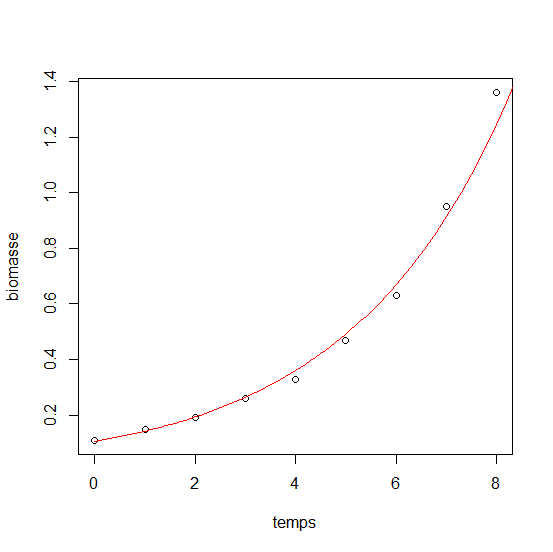
Avec : et

Pour tracer la courbe de régression :

> alfa = exp(coef(reg\_exp)[1])

> beta = coef(reg\_exp)[2]

> curve(alfa\*exp(beta\*x) , from=0 , to=10, col = "red", add = TRUE)



**Régression logarithmique**

La capacité d’oxygénation d’un fermenteur (O2L) est déterminée en suivant la cinétique de transfert d’oxygène en absence de microorganismes selon le tableau suivant :



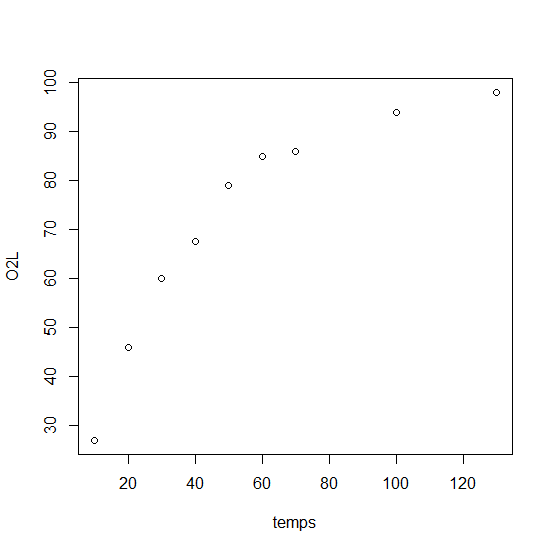
Tracer le nuage de point de O2Li en fonction du temps

> temps = c(10, 20, 30, 40, 50, 60, 70, 100, 130)

> O2L = c(27, 46, 60, 67.5, 79, 85, 86, 94, 98)

> plot(temps, O2L)

>



Il est clair qu’il s’agit d’une régression logarithmique. Pour trouver le model de cette régression nous allons appliquer la fonction lm() à exp(O2L) et temps :

> reg\_log = lm(O2L ~ log(temps))

> reg\_log

Call:

lm(formula = O2L ~ log(temps))

Coefficients:

(Intercept) log(temps)

-38.98 29.12

> summary(reg\_log)

Call:

lm(formula = O2L ~ log(temps))

Residuals:

Min 1Q Median 3Q Max

-4.7431 -1.1041 -0.9254 1.2809 4.7691

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) -38.980 5.454 -7.147 0.000186 \*\*\*

log(temps) 29.116 1.411 20.634 1.58e-07 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.202 on 7 degrees of freedom

Multiple R-squared: 0.9838, Adjusted R-squared: 0.9815

F-statistic: 425.8 on 1 and 7 DF, p-value: 1.576e-07

>

Selon ce model :

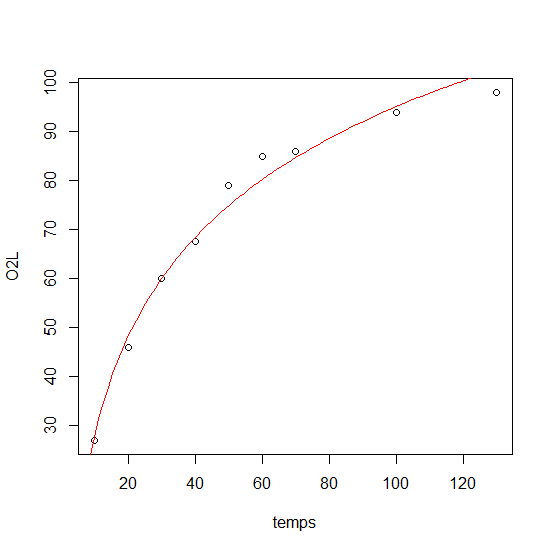
Avec : et

Pour tracer la courbe de régression :

> a=reg\_log$coef[1]

> b=reg\_log$coef[2]

> curve(a+ b\*log(x) , from=0 , to=140, col = "red", add = TRUE)



**Régression polynomiale**

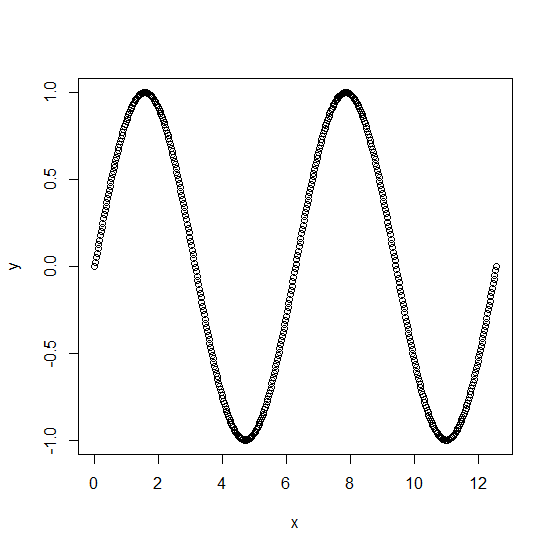
Dans cette partie nous allons présenter un modèle de régression polynomiale. Pour cela nous allons créer le nuage de points suivant :

> x= seq(0, 4\*pi, length=500)

> y=sin(x)

>

> plot(x,y)

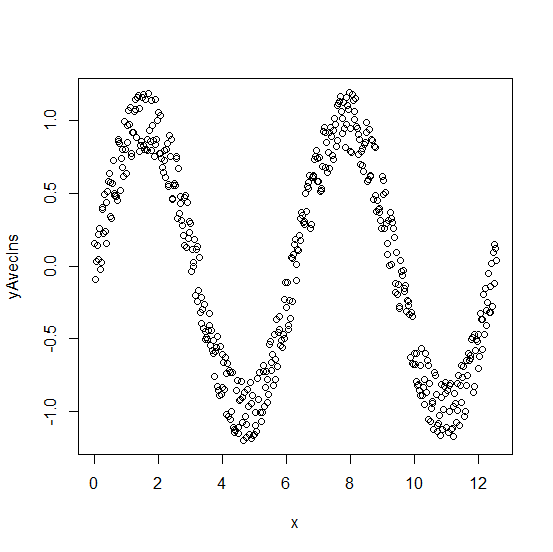


Pour avoir un nuage de point qui ressemble à des données expérimentales nous allons ajouter un brui (une incertitude)

> yIns= sample(y/5, 500)

> yAvecIns = y + yIns

> plot(x,yAvecIns)



Il est clair qu’il s’agit d’une régression polynomiale

Nous allons chercher un polynôme de régression de degré 5 parce que le nuage de points à 2 max locaux et 2 min locaux

> reg\_poly = lm(yAvecIns ~ x + I(x^2) + I(x^3) + I(x^4) + I(x^5))

> reg\_poly

Call:

lm(formula = yAvecIns ~ x + I(x^2) + I(x^3) + I(x^4) + I(x^5))

Coefficients:

(Intercept) x I(x^2) I(x^3) I(x^4) I(x^5)

-0.614740 3.578164 -2.292867 0.510208 -0.046465 0.001483

> summary(reg\_poly)

Call:

lm(formula = yAvecIns ~ x + I(x^2) + I(x^3) + I(x^4) + I(x^5))

Residuals:

Min 1Q Median 3Q Max

-0.6832 -0.1991 0.0109 0.1986 0.7707

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) -0.6147404 0.0694442 -8.852 <2e-16 \*\*\*

x 3.5781644 0.1120601 31.931 <2e-16 \*\*\*

I(x^2) -2.2928670 0.0553980 -41.389 <2e-16 \*\*\*

I(x^3) 0.5102078 0.0111900 45.595 <2e-16 \*\*\*

I(x^4) -0.0464647 0.0009820 -47.314 <2e-16 \*\*\*

I(x^5) 0.0014828 0.0000311 47.677 <2e-16 \*\*\*

---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.2633 on 494 degrees of freedom

Multiple R-squared: 0.8687, Adjusted R-squared: 0.8674

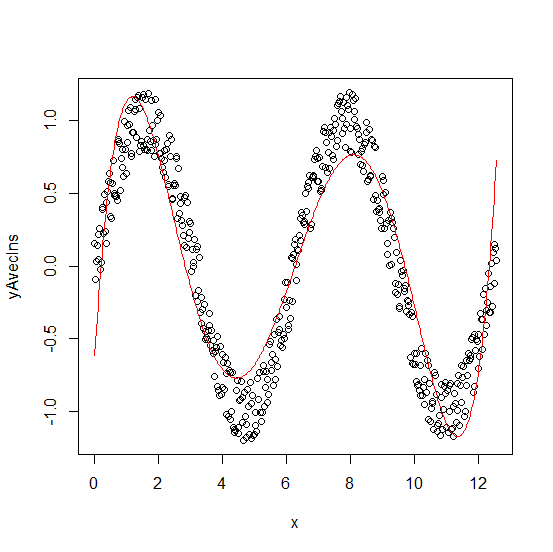
F-statistic: 653.7 on 5 and 494 DF, p-value: < 2.2e-16

>

Selon ce model

Pour tracer la courbe de régression :

> lines (x, fitted(reg\_poly) , col = "red" )



**Un Autre exemple de régression polynomial**

**Création des données (vecteurs X et Y)**

> x=seq(0,100)

> X=sample(x,100, rep = T)

> X=sample(X,100, rep = T)

> X=sample(X,100, rep = T)

> y=sin(X\*pi/50)

> Y=5\*y+rnorm(100)

> plot(X,Y)

**Création du model polynomial**

> model = lm(Y ~ X + I(X^2)+ I(X^3)+ I(X^4)+ I(X^5))

> beta = coef(model)

> curve(beta[1] + beta[2]\*x + beta[3]\*x^2 + beta[4]\*x^3 + beta[5]\*x^4 + beta[6]\*x^5 , col = "red", add = T)

