**Linear Regression**

In statistics, linear regression is a linear approach to modeling the relationship between a scalar response (or *dependent variable*) and one or more explanatory variables (or *independent variables*). The relationships are modeled using **linear basis functions**, essentially replacing each input with a function of the input. **This** is linear regression:

\[Y = \alpha + \beta\_1 f\_1(X) + \beta\_2 f\_2(X) + … + \beta\_n f\_n(X) + \epsilon\]

This is only a subclass of linear regression:

\[Y = \alpha + \beta\_1 X\_1 + \beta\_2 X\_1 + … + \beta\_n X\_n+ \epsilon\]

This is linear regression as well:

\[Y = \alpha + \beta\_1 X\_1^2 + \beta\_2 log(X\_1) + … + \beta\_n sin(X\_n)+ \epsilon\]

**Estimation**

In R, the lm function is used to fit linear models. For panel data, the plm function from the plm package can be used.

**Exercise** Simulate an exponential growth model \(y(t) = y\_0e^{kt}\) and estimate the growth rate \(k\) and the initial population \(y\_0\).

# time grid

t <- seq(0, 10, by = 0.01)

# simulate y values for k = 0.33 and initial population y0 = 1000

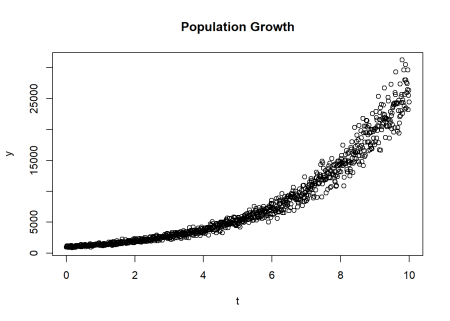
y <- 1000\*exp(0.33\*t)

# add random noise

y <- y \* rnorm(n = length(y), mean = 1, sd = 0.1)

# plot

plot(y ~ t, main = "Population Growth")



Assume the \(y\) values generated above are given. We don’t know the initial population \(y0\) nor the growth rate \(k\). To estimate these parameters we proceed as follows:

\[z = ln(y(t)) = ln(y\_0e^{kt}) = ln(y\_0) + k\;t = \alpha + \beta\;t\]  
where \(\alpha=ln(y\_0)\) and \(\beta=k\).

# transform the output variable

z <- log(y)

# fit the model

mod <- lm(z ~ t)

# extract the coefficients

mod.c <- coefficients(mod)

# extract alpha

alpha <- mod.c[1]

# extract beta

beta <- mod.c[2]

# compute y0

y0 <- exp(alpha)

# compute k

k <- beta

# print estimates

sprintf("y0 = %s; k = %s", y0, k)

## [1] "y0 = 997.365557000044; k = 0.329840311556089"

The estimates seems close to the true values \(y\_0=1000\) and \(k = 0.33\), but how can we test for them to be equal? We need confidence intervals.

# computes confidence intervals for the parameters in the model

mod.i <- confint(mod, level = 0.95)

## 2.5 % 97.5 %

## (Intercept) 6.8927301 6.9175046

## t 0.3276953 0.3319853

The true value of \(k = \beta = 0.33\) is inside the confidence interval obtained above and has been consistently estimated. To check for \(y\_0\) we need to transform the confidence interval obtained for \(\alpha\).

# compute the confidence interval for y0

low <- exp(mod.i[1,1])

upp <- exp(mod.i[1,2])

# print

sprintf("Confidence interval for y0: %s - %s", round(low,1), round(upp,1))

## [1] "Confidence interval for y0: 985.1 - 1009.8"

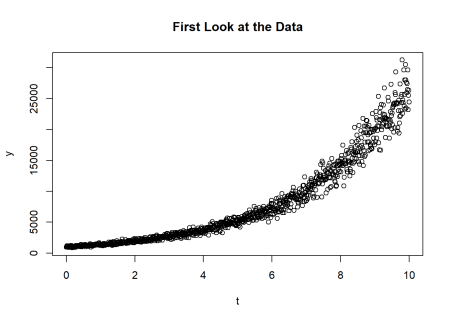
**Model Selection**

In the previous example we knew the functional form linking the inputs to the output variable. This is not often the case in economics and finance, where the model is not known a priori and has to be deduced from the data.

**Exercise** Repeat the same exercise of the previous section but assume no model is given a priori. Deduce a reasonable model and estimate its parameters.

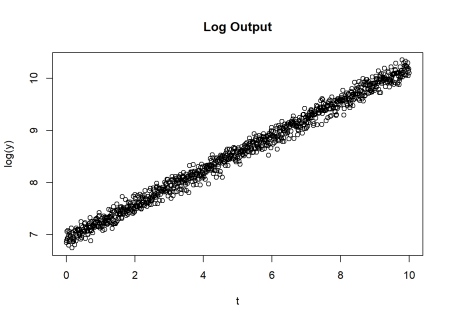
# visualize the data

plot(y ~ t, main = "First Look at the Data")



The data are not linear with respect to \(t\). They seem to be some exponential, quadratic, cubic… function of \(t\). We can try to take the log of \(y\) and see what they look like.

plot(log(y) ~ t, main = "Log Output")



Much better! This seems linear but we want to test also for quadratic and cubic effects. Build the full model:

\[ln(y) = \alpha + \beta\_1 t + \beta\_2 t^2 + \beta\_3 t^3+ \epsilon\]  
and fit it to the data.

# build a data frame of regressors

data <- data.frame(log.y = log(y), t1 = t, t2 = t^2, t3 = t^3)

# fit the model

mod <- lm(log.y ~ t1 + t2 + t3, data = data)

# summary statistics

summary(mod)

##

## Call:

## lm(formula = log.y ~ t1 + t2 + t3, data = data)

##

## Residuals:

## Min 1Q Median 3Q Max

## -0.32659 -0.06253 0.00485 0.06780 0.28404

##

## Coefficients:

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

## (Intercept) 6.908e+00 1.260e-02 548.241 <2e-16 \*\*\*

## t1 3.272e-01 1.092e-02 29.972 <2e-16 \*\*\*

## t2 6.196e-04 2.538e-03 0.244 0.807

## t3 -3.954e-05 1.668e-04 -0.237 0.813

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## Residual standard error: 0.1 on 997 degrees of freedom

## Multiple R-squared: 0.9891, Adjusted R-squared: 0.9891

## F-statistic: 3.029e+04 on 3 and 997 DF, p-value: < 2.2e-16

From the output we discover that:

* only the intercept (\(\alpha\)) and t1 (\(\beta\_1\)) are statistically different from zero. The probability for them to be zero is infact less than \(10^{-16}\).
* t2 and t3 are not statistically different from zero. The probability of observing such estimates if their true value is zero, is infact pretty high: around 80%. We cannot reject the hypothesis of \(\beta\_2\) and \(\beta\_3\) to be zero and we are going to accept it.
* the R-squared is close to 1: the model is able to capture almost all the variability in the data

Since \(\beta\_2\) and \(\beta\_3\) are not statistically different from zero, we reduce the full model and estimate it again.

\[ln(y) = \alpha + \beta\_1 t+ \epsilon\]

# fit the model

mod <- lm(log.y ~ t1, data = data)

# summary statistics

summary(mod)

##

## Call:

## lm(formula = log.y ~ t1, data = data)

##

## Residuals:

## Min 1Q Median 3Q Max

## -0.32628 -0.06206 0.00441 0.06815 0.28363

##

## Coefficients:

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

## (Intercept) 6.905117 0.006312 1093.9 <2e-16 \*\*\*

## t1 0.329840 0.001093 301.8 <2e-16 \*\*\*

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## Residual standard error: 0.09993 on 999 degrees of freedom

## Multiple R-squared: 0.9891, Adjusted R-squared: 0.9891

## F-statistic: 9.106e+04 on 1 and 999 DF, p-value: < 2.2e-16

To understand the meaning of the estimated coefficients, we proceed as follows:

\[ln(y) = \alpha + \beta\_1 t \rightarrow y = exp(\alpha + \beta\_1 t) = e^{\alpha}e^{\beta\_1t}=y\_0e^{k t}\]  
where:

# extract estimates

mod.c <- coef(mod)

# y0

y0 <- exp(mod.c[1])

# k

k <- mod.c[2]

# print

sprintf("y0 = %s; k = %s", y0, k)

## [1] "y0 = 997.365557000044; k = 0.329840311556089"

**R-squared**

R-squared is a goodness-of-fit measure for linear regression models. This statistic indicates the percentage of the variance in the dependent variable that the independent variables explain collectively. R-squared measures the strength of the relationship between your model and the dependent variable on a convenient 0 – 100% scale.

A good predictive model should achieve high values of R-squared, while this measure plays no role when assessing the significancy of the parameters.

**Exercise** Simulate a dataset from the model \(y = 2sin(x) + 1\) and see how the R-square changes when increasing the noise in the data. Is the significance of the estimates affected?

# x grid

x <- seq(0, 2\*pi, by = 0.01)

# y

y = 2\*sin(x)+1

# y: low noise

y.low <- y + rnorm(n = length(y), mean = 0, sd = 0.1)

# y: medium noise

y.mid <- y + rnorm(n = length(y), mean = 0, sd = 1)

# y: high noise

y.high <- y + rnorm(n = length(y), mean = 0, sd = 10)

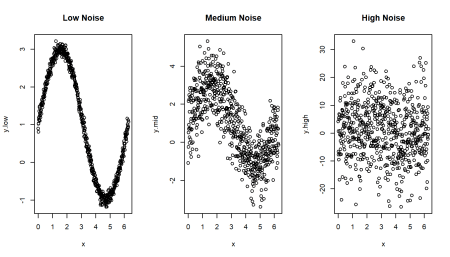
# plot

layout(t(1:3))

plot(y.low ~ x, main = "Low Noise")

plot(y.mid ~ x, main = "Medium Noise")

plot(y.high ~ x, main = "High Noise")



# low noise

summary(lm(y.low ~ sin(x)))

##

## Call:

## lm(formula = y.low ~ sin(x))

##

## Residuals:

## Min 1Q Median 3Q Max

## -0.27887 -0.06682 0.00323 0.06331 0.33386

##

## Coefficients:

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

## (Intercept) 1.004789 0.003924 256.0 <2e-16 \*\*\*

## sin(x) 1.995093 0.005553 359.3 <2e-16 \*\*\*

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## Residual standard error: 0.09842 on 627 degrees of freedom

## Multiple R-squared: 0.9952, Adjusted R-squared: 0.9952

## F-statistic: 1.291e+05 on 1 and 627 DF, p-value: < 2.2e-16

# medium noise

summary(lm(y.mid ~ sin(x)))

##

## Call:

## lm(formula = y.mid ~ sin(x))

##

## Residuals:

## Min 1Q Median 3Q Max

## -3.0789 -0.6506 -0.0113 0.7228 2.8477

##

## Coefficients:

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

## (Intercept) 0.97263 0.03964 24.53 <2e-16 \*\*\*

## sin(x) 2.08865 0.05610 37.23 <2e-16 \*\*\*

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## Residual standard error: 0.9943 on 627 degrees of freedom

## Multiple R-squared: 0.6886, Adjusted R-squared: 0.6881

## F-statistic: 1386 on 1 and 627 DF, p-value: < 2.2e-16

# high noise

summary(lm(y.high ~ sin(x)))

##

## Call:

## lm(formula = y.high ~ sin(x))

##

## Residuals:

## Min 1Q Median 3Q Max

## -28.8317 -6.3656 -0.1938 6.7277 29.8982

##

## Coefficients:

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

## (Intercept) 1.0089 0.3936 2.563 0.0106 \*

## sin(x) 2.3579 0.5570 4.233 2.65e-05 \*\*\*

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## Residual standard error: 9.873 on 627 degrees of freedom

## Multiple R-squared: 0.02779, Adjusted R-squared: 0.02624

## F-statistic: 17.92 on 1 and 627 DF, p-value: 2.648e-05

The R-squared is almost 100% for y.low, 68% for y.mid and only 3% for y.high. In the first case, we are able to predict y based on x with very high accuracy. In the second case the accuracy drops. In the third case we have basically no predictive power but we were able to assess the statistically significant impact of \(sin(x)\) on \(y\). On the other hand, the uncertainty associated with the estimates of the coefficients increased and the significancy levels drop. For even higher noise levels we won’t be able to assess the statistically significant impact of the regressor on the response variable, but this problem can be solved increasing the number of observations when possible (try as an exercise).

After running a regression analysis, we should check if the model works well for data. We paid attention to regression results, such as slope coefficients, p-values, or R-squared but that’s not the whole picture. **Residuals** could show how poorly a model represents data. Residuals are leftover of the outcome variable after fitting a model (predictors) to data and they could reveal unexplained patterns in the data by the fitted model. Using this information, not only could we check if linear regression assumptions are met, but we could improve our model in an exploratory way.  
Refer to: [Understanding Diagnostic Plots for Linear Regression Analysis](https://data.library.virginia.edu/diagnostic-plots/).

**Testing CAPM**

\[E[R\_i – r\_f] = \beta\_i E[R\_{mkt} – r\_f]\]  
where:

* \(R\_{i\_t}\): return on asset \(i\) at time \(t\)
* \(r\_f\): risk-free return at time \(t\)
* \(R\_{m,t}\): return on the market portfolio at time \(t\)

To test the model we use the following [data file](https://storage.guidotti.dev/course/asset-pricing-unine-2019-2020/basic-linear-regressions-for-finance.csv) containing stock data from the website of Kenneth R. French. It includes the monthly simple computed stock returns in percentage points for decile portfolios formed on beta over the period 1963-2017. These are total returns (i.e. they include dividends).

# read data

data <- read.csv('<https://storage.guidotti.dev/course/asset-pricing-unine-2019-2020/basic-linear-regressions-for-finance.csv>')

# drop date

data <- data[,-1]

# print

head(data)

## Lo.10 Dec.2 Dec.3 Dec.4 Dec.5 Dec.6 Dec.7 Dec.8 Dec.9 Hi.10 Mkt.RF RF

## 1 1.35 0.77 0.08 -0.24 -0.69 -1.20 -0.49 -1.39 -1.94 -0.77 -0.39 0.27

## 2 3.52 3.89 4.29 5.25 5.23 7.55 7.57 4.91 9.04 10.47 5.07 0.25

## 3 -3.09 -2.24 -0.54 -0.97 -1.37 -0.27 -0.63 -1.00 -1.92 -3.68 -1.57 0.27

## 4 1.25 -0.12 2.00 5.12 2.32 1.78 6.63 4.78 3.10 3.01 2.53 0.29

## 5 -0.91 -0.15 1.60 -2.05 -0.94 -0.69 -1.32 -0.51 -0.20 0.52 -0.85 0.27

## 6 3.86 0.63 2.31 1.83 3.00 2.36 1.25 3.45 0.30 1.28 1.83 0.29

# get the portfolios

portfolios <- data[,-c(11,12)]

# compute excess returns

portfolios <- portfolios - data$RF

# print

head(portfolios)

## Lo.10 Dec.2 Dec.3 Dec.4 Dec.5 Dec.6 Dec.7 Dec.8 Dec.9 Hi.10

## 1 1.08 0.50 -0.19 -0.51 -0.96 -1.47 -0.76 -1.66 -2.21 -1.04

## 2 3.27 3.64 4.04 5.00 4.98 7.30 7.32 4.66 8.79 10.22

## 3 -3.36 -2.51 -0.81 -1.24 -1.64 -0.54 -0.90 -1.27 -2.19 -3.95

## 4 0.96 -0.41 1.71 4.83 2.03 1.49 6.34 4.49 2.81 2.72

## 5 -1.18 -0.42 1.33 -2.32 -1.21 -0.96 -1.59 -0.78 -0.47 0.25

## 6 3.57 0.34 2.02 1.54 2.71 2.07 0.96 3.16 0.01 0.99

**Time-Series Approach**

The time-series approach consists in the following regression:

\[R\_{i,t} – r\_f = \alpha\_i + \beta\_i (R\_{m,t} – r\_f)+ \epsilon\_{i,t}\]  
i.e.

\[Y\_{i,t} = \alpha\_i + \beta\_i X\_t + \epsilon\_{i,t}\]

where:

* \(R\_{i\_t}\): return on asset \(i\) at time \(t\)
* \(r\_f\): risk-free return at time \(t\)
* \(R\_{m,t}\): return on the market portfolio at time \(t\)
* \(Y\_{i,t} = R\_{i,t} – r\_f\): excess return on asset \(i\) at time \(t\)
* \(X\_t = R\_{mkt}-r\_f\): excess return on the market portfolio at time \(t\)

The CAPM implies \(\alpha\_i = 0\). Infact, if \(\alpha\_i \neq 0\) then taking the expectation on both terms of the equation violates the CAPM.

\[E[R\_{i,t} – r\_f] = E[\alpha\_i + \beta\_i (R\_{m,t} – r\_f)] = \alpha\_i + \beta\_i E[R\_{m,t} – r\_f] \neq \beta\_i E[R\_{m,t} – r\_f]\]

Therefore, the CAPM is rejected if we obsrve \(\alpha\) statistically different from zero.

# define an empty data frame

capm <- data.frame()

# define a matrix to store residuals

eps <- matrix(NA, nrow = nrow(portfolios), ncol = ncol(portfolios))

# for each portfolio...

for(i in 1:ncol(portfolios)){

# linear regression

mod <- lm(portfolios[,i] ~ data$Mkt.RF)

# summary

mod.s <- summary(mod)

# store residuals

eps[,i] <- residuals(mod)

# extract coefficients

alpha <- mod.s$coefficients[1,'Estimate']

beta <- mod.s$coefficients[2,'Estimate']

# extract standard errors of the estimates

sd.alpha <- mod.s$coefficients[1,'Std. Error']

sd.beta <- mod.s$coefficients[2,'Std. Error']

# compute the average excess return

excess <- mean(portfolios[,i])

# store everything into the capm dataframe

row <- c(excess, alpha, sd.alpha, beta, sd.beta)

capm <- rbind(capm, row)

}

# assign colnames

colnames(capm) <- c('', 'alpha', 'sd.alpha', 'beta', 'sd.beta')

# print

capm

## alpha sd.alpha beta sd.beta

## 1 0.5465291 0.219840960 0.08358282 0.6152566 0.01892420

## 2 0.5221713 0.131098193 0.07708074 0.7365138 0.01745205

## 3 0.5882875 0.145659989 0.06624880 0.8336070 0.01499956

## 4 0.6657951 0.149145083 0.06207395 0.9730148 0.01405432

## 5 0.5541590 0.013151107 0.05977863 1.0188884 0.01353463

## 6 0.6346483 0.059530018 0.06389041 1.0831290 0.01446559

## 7 0.5194801 -0.095702502 0.07022164 1.1585827 0.01589906

## 8 0.6728287 -0.005224589 0.08177222 1.2769881 0.01851426

## 9 0.6400306 -0.098993437 0.10113883 1.3918151 0.02289910

## 10 0.6306269 -0.224398053 0.13376284 1.6102814 0.03028559

We estimated \(\alpha\_i\) for all the ten portfolios and their standard errors. Each \(\alpha\_i\) is (approximately) normally distributed with standard deviation \(\sigma\_{\alpha\_i}\). Therefore to test if all \(\alpha\) are jointly equal to zero we can define the following random variable

\[\chi^2\_N=\sum\_{i=1}^N \Bigl(\frac{\alpha\_i-0}{\sigma\_{\alpha\_i}}\Bigl)^2\]

which is the sum of \(N\) (approximately) independent standard normal variables, i.e. it has a (approximate) chi-squared distribution with \(N\) degrees of freedom.

# chi squared random variable

chi.sq <- sum((capm$alpha/capm$sd.alpha)^2)

## [1] 26.96824

Which is the probability of observing a value equal or greater than 26.9682402 if it has a chi-squared distribution with ten degrees of freedom?

pchisq(q = chi.sq, df = nrow(capm), lower.tail = FALSE)

## [1] 0.002634639

The CAPM would be rejected at a confidence level of 99%. The problem is that \(cov(\alpha\_i,\alpha\_j)\) will not be zero. Thus, it is common to use \(\boldsymbol \alpha^\intercal cov(\boldsymbol \alpha)^{-1} \boldsymbol \alpha\). Now we follow this approach to take correlation into account and compute the following statistic (GRS Test), which follows an F distributions assuming normally distributed error terms:

\[f\_{GRS} \sim F(n,\tau – n – k)=\frac{\tau-n-k}{n}\frac{\hat{\alpha}^\intercal\hat\Omega^{-1}\hat\alpha}{1+\hat\mu\_f^\intercal\hat\Sigma^{-1}\_f\hat\mu\_f}\]  
where:

* \(T\): number of time perdiods
* \(n\): number of assets
* \(k\): number of factors (in our case 1)
* \(\alpha\): vector of estimated \(\alpha\_i\)
* \(\Omega\): covariance matrix of residuals
* \(\mu\): vector giving the sample means of the factor(s)
* \(\Sigma\): covariance matrix of factors (in our case it reduces to the variance of the market excess return)

# number of time perdiods

t <- nrow(portfolios)

# number of assets

n <- ncol(portfolios)

# number of factors (in our case 1)

k <- 1

# vector of estimated alpha\_i

alpha <- capm$alpha

# covariance matrix of residuals

omega <- cov(eps)

# vector giving the sample means of the factor

mu <- mean(data$Mkt.RF)

# covariance matrix of factors

sigma <- var(data$Mkt.RF)

# F-statistic (GRS test)

f <- (t-n-k)/n \* (alpha %\*% solve(omega) %\*% alpha)/(1 + mu %\*% solve(sigma) %\*% mu)

# p-value

pf(q = f, df1 = n, df2 = t-n-1, lower.tail = FALSE)

## [,1]

## [1,] 0.03102946

The CAPM is still rejected at a confidence level of 95%, even wen taking into account the correlations between \(\alpha\_i\).

Finally, dropping the assumption of normally distributed error terms and taking correlation into account as well, there exists a test-statistic that asymptotically approaches the \(\chi^2\) distribution:

\[J \sim \chi^2(n)=\tau\frac{\hat{\alpha}^\intercal\hat\Omega^{-1}\hat\alpha}{1+\hat\mu\_f^\intercal\hat\Sigma^{-1}\_f\hat\mu\_f}\]

# chi squared statistic

x <- t \* (alpha %\*% solve(omega) %\*% alpha)/(1 + mu %\*% solve(sigma) %\*% mu)

# p-value

pchisq(q = x, df = n, lower.tail = FALSE)

## [,1]

## [1,] 0.02617805

The CAPM is still rejected at a confidence level of 95%, even when taking into account the non-normality of error terms together with correlation of \(\alpha\_i\).

We now consider a different approach to test CAPM. Note: what is done below is essentially the same of using dummy variables. Consider the model:

\[R\_{i,t} – r\_f = \alpha + \sum\_{j=1}^N\beta\_j \delta\_{i,j}(R\_{m,t} – r\_f)+ \epsilon\_{i,t}\]

where \(\delta\_{i,j}\) is the Kronecker delta, i.e

\[\delta\_{i\_j} =  
\begin{cases}  
1, & \text{if } i=j,\\  
0, & \text{if } i\neq j.  
\end{cases}\]

The model correctly reduces to the standard CAPM for each asset \(i\). For example, consider the first asset \(i=1\):

\[R\_{1,t} – r\_f = \alpha + \sum\_{j=1}^N\beta\_j \delta\_{1,j}(R\_{m,t} – r\_f)+ \epsilon\_{1,t}\]  
Now, \(\delta\_{1,j}\) equals 1 only for \(j=1\) and vanishes for all other terms. The only term which contributes to the summation is therefore \(j=1\) and we have the standard CAPM for the first asset, which predicts \(\alpha=0\):

\[R\_{1,t} – r\_f = \alpha + \beta\_1 (R\_{m,t} – r\_f)+ \epsilon\_{1,t}\]

We can repeat the procedure for all assets and we obtain the standard CAPM for all assets, where now \(\alpha\) is a common parameter, equal to 0 according to CAPM. Test for \(\alpha=0\) and we will test for CAPM to hold.

# number of assets

n.p <- ncol(portfolios)

# number of observations for each asset

n.t <- nrow(portfolios)

# matrix of excess returns and the n.p regressors (delta\_{i,j} \* (R\_{m,t} - r\_f))

M <- matrix(0, nrow = n.t\*n.p, ncol = n.p+1)

colnames(M) <- c('excess', colnames(portfolios))

# fill the first column with the excess returns

M[,1] <- unlist(portfolios)

# fill each column with (R\_{m,t} - r\_f) only if i==j

for(i in 1:n.p){

M[1:n.t + (i-1)\*n.t, i+1] <- data$Mkt.RF

}

# linear regression

mod <- lm(excess ~ ., data = as.data.frame(M))

# summary

summary(mod)

##

## Call:

## lm(formula = excess ~ ., data = as.data.frame(M))

##

## Residuals:

## Min 1Q Median 3Q Max

## -14.8914 -1.0903 -0.0251 1.0637 13.0585

##

## Coefficients:

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

## (Intercept) 0.02941 0.02622 1.122 0.262

## Lo.10 0.62044 0.01865 33.270 <2e-16 \*\*\*

## Dec.2 0.73928 0.01865 39.643 <2e-16 \*\*\*

## Dec.3 0.83677 0.01865 44.870 <2e-16 \*\*\*

## Dec.4 0.97627 0.01865 52.351 <2e-16 \*\*\*

## Dec.5 1.01845 0.01865 54.612 <2e-16 \*\*\*

## Dec.6 1.08395 0.01865 58.125 <2e-16 \*\*\*

## Dec.7 1.15518 0.01865 61.944 <2e-16 \*\*\*

## Dec.8 1.27605 0.01865 68.426 <2e-16 \*\*\*

## Dec.9 1.38832 0.01865 74.446 <2e-16 \*\*\*

## Hi.10 1.60337 0.01865 85.978 <2e-16 \*\*\*

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## Residual standard error: 2.105 on 6529 degrees of freedom

## Multiple R-squared: 0.8421, Adjusted R-squared: 0.8419

## F-statistic: 3482 on 10 and 6529 DF, p-value: < 2.2e-16

Note that all \(\beta\_i\) are the same of those estimated independently, while the intercept is not statistically significant, i.e. \(\alpha\) is not statistically different from zero. The CAPM cannot be rejected. Note that when performing this kind of tests the reverse does not hold: we cannot say that based on this test the CAPM holds. Infact, we caould have observed \(\alpha\) not statistically different from zero both because:

* the true value of \(\alpha\) is zero
* we don’t have enough data and the uncertainty of the parameters is too high to detect the significant difference between the true \(\alpha\) and zero. In other words, we didn’t have enough *statistical power* to tell the difference between zero and somthing close to zero. Increasing the size of the dataset would allow us to estimare a significant \(\alpha \neq0\)

What do we learn from this? First, **not rejecting an hypothesis does not mean accepting it**, otherwise the last apprach would contradict the previous ones. Second, for the same puropose there can be many different approaches, more or less suited to it, and several tests with different statistical power, i.e. able to distinguish better between the true value and something close to the true value.

**Cross-Sectional Approach**

The cross-sectional approach consists in the following regression:

\[E[R\_i – r\_f] = \beta\_i E[R\_{mkt} – r\_f]\]

i.e.

\[Y\_i = \lambda X\_i + \theta + \epsilon\_{i}\]

where:

* \(Y\_i=E[R\_i-r\_f]\): average excess return on asset \(i\)
* \(X\_i=\beta\_i\): coefficients estimated in the time-series approach on asset \(i\)

The CAPM implies \(\gamma=E[R\_{mkt}-r\_f]\) and \(\theta=0\). Infact, if \(\lambda \neq E[R\_{mkt}-r\_f]\) and/or \(\theta \neq 0\) then:

\[E[R\_i-r\_f]= Y\_i = \lambda X\_i = \lambda \beta\_i +\theta \neq \beta\_i E[R\_{mkt} – r\_f]\]

Therefore, the CAPM is rejected if we obsrve \(\lambda\) statistically different from \(E[R\_{mkt}-r\_f]\) and/or \(\theta \neq 0\).

# linear regression

sml <- lm(capm$`` ~ capm$beta)

# print

summary(sml)

##

## Call:

## lm(formula = capm$`` ~ capm$beta)

##

## Residuals:

## Min 1Q Median 3Q Max

## -0.087237 -0.034292 0.002738 0.030721 0.078437

##

## Coefficients:

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

## (Intercept) 0.48585 0.06353 7.647 6.03e-05 \*\*\*

## capm$beta 0.10432 0.05734 1.819 0.106

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## Residual standard error: 0.05231 on 8 degrees of freedom

## Multiple R-squared: 0.2927, Adjusted R-squared: 0.2043

## F-statistic: 3.31 on 1 and 8 DF, p-value: 0.1063

We estimated \(\theta\) statistically different from zero and the CAPM is rejected.  
Regarding \(\lambda\), we estimated a value of 0.1043244. Is it statistically different from \(E[R\_{mkt} – r\_f]\)?

# mean excess return on the market portfolio

mean(data$Mkt.RF)

## [1] 0.5309786

# confidence intervals at 95%

confint(sml, level = 0.95)

## 2.5 % 97.5 %

## (Intercept) 0.33934005 0.6323572

## capm$beta -0.02790093 0.2365497

The mean excess return does not follow inside the confidence interval: \(\lambda\) is statistically different from \(E[R\_{mkt} – r\_f]\). The CAPM is rejected.

To conclude, we represent graphically the results obtained.

# grid of beta

betas <- seq(0, 2, by = 0.01)

# excess returns by CAPM

E.R <- betas \* mean(data$Mkt.RF)

# plot

plot(E.R ~ betas, type = 'l', lwd = 2, col = 'orange',

main = "SML vs Beta Regression", xlab = 'Beta',

ylab = 'Mean Excess Return')

# add points estimated in the time-series approach

points(x = capm$beta, y = capm$``, pch = 16, cex = 1)

text(labels = 1:10, x = capm$beta, y = capm$``, cex = 1, pos = 3)

# add regression line

abline(sml, lty = 'dashed')

