STATISTICAL MODELLING Part IIC, Michaelmas 2019
Practical 2: More on the basics of R By courtesy of Dr. S. A. Bacallado and Dr. R. Shah

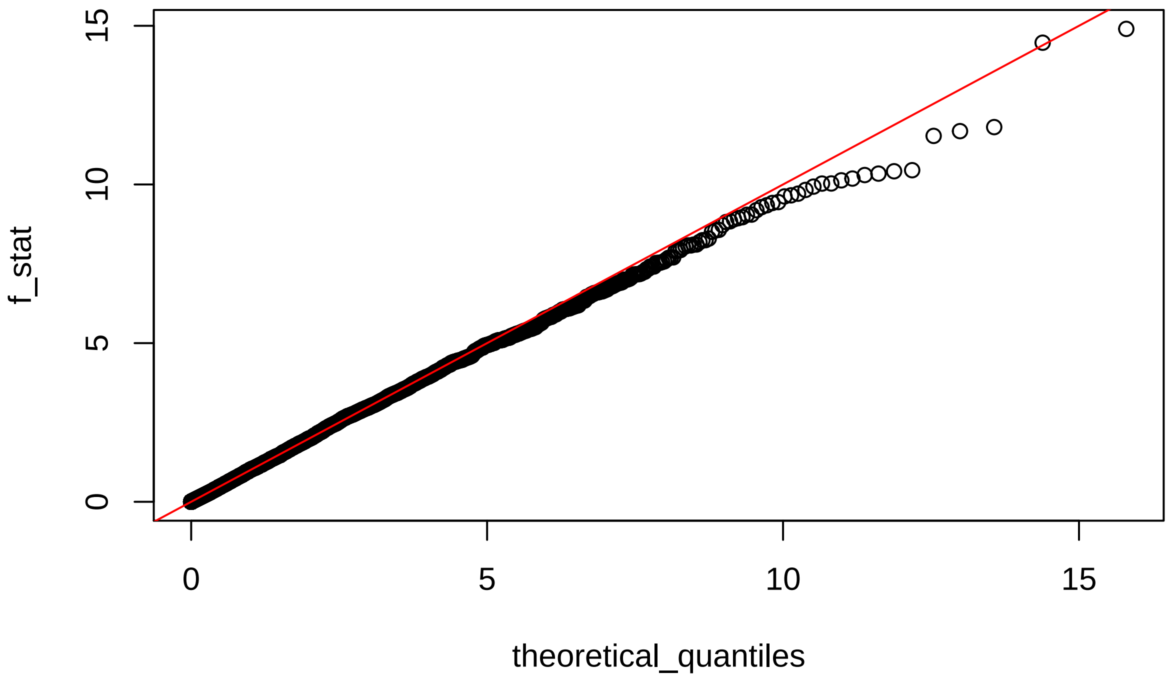
Solutions to exercises

Once you define the functions `LinMod_sim` and `qqplot_F`, we can run a simulation with the purpose of checking whether t statistics in a normal linear model have a t_{n-p} distribution.

```
# Let us fix the "random" numbers, so that we all get the same results
set.seed(1)
# Set the number of predictors and observations
p=10
n=200
# Sample a design matrix with Uniform(0,1) entries
X = matrix(runif(n*p),n,p)
# Compute t-statistics in 10000 simulations from the model
t_stat_mat1 <- LinMod_sim(X)
```

If the t statistics have a t_{n-p} distribution, then the squared t statistics have an $F_{1,n-p}$ distribution, indeed

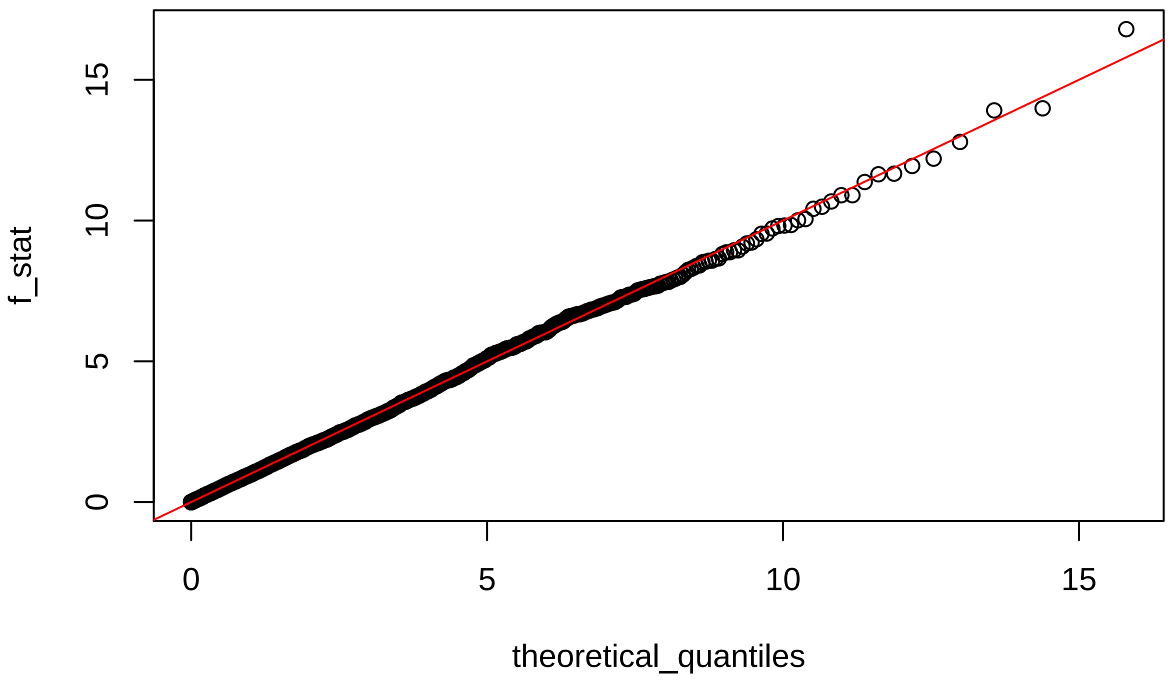
```
# Compute square t statistics for the first coefficient
f <- t_stat_mat1[1,]^2
qqplot_F(f,1,n-p)
```



```
## [1] 0.9514
```

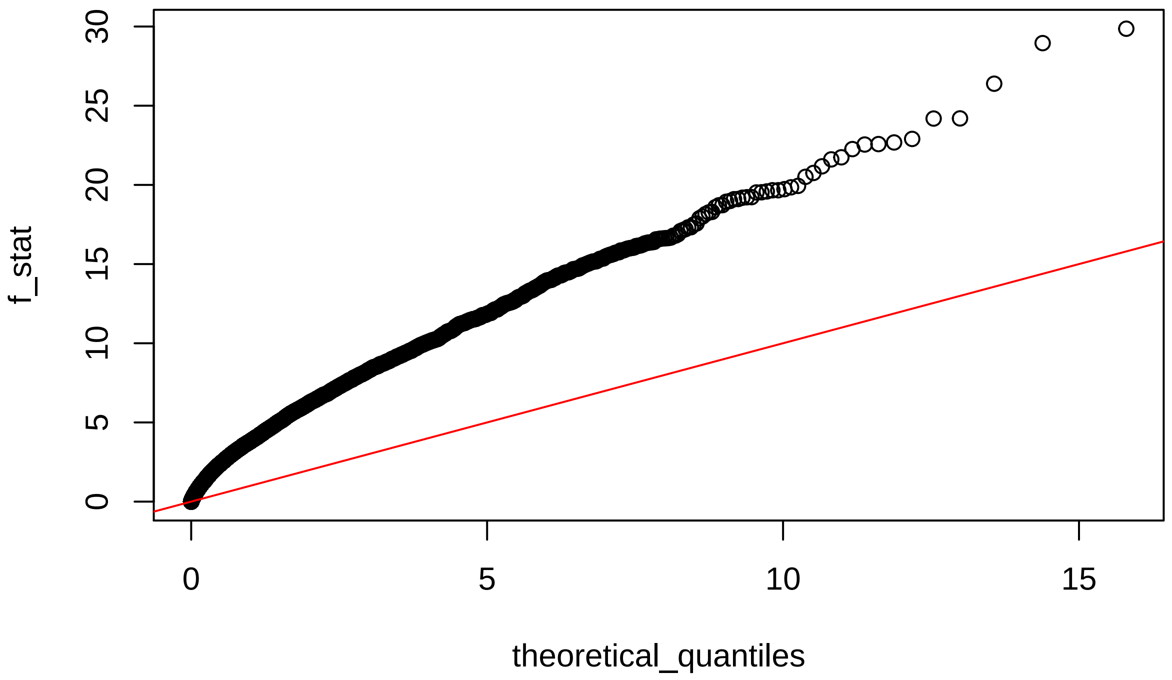
Now, let's see what happens if we make some of the true coefficients different from 0.

```
# Compute t-statistics in 10000 simulations from the new model
t_stat_mat2 <- LinMod_sim(X,Beta=c(0,rnorm(p-1)))
# QQ plot for the first coefficient
f <- t_stat_mat2[1,]^2
qqplot_F(f,1,n-p)
```



```
## [1] 0.9516
```

```
# QQ plot for the second coefficient
f <- t_stat_mat2[2,]^2
qqplot_F(f,1,n-p)
```

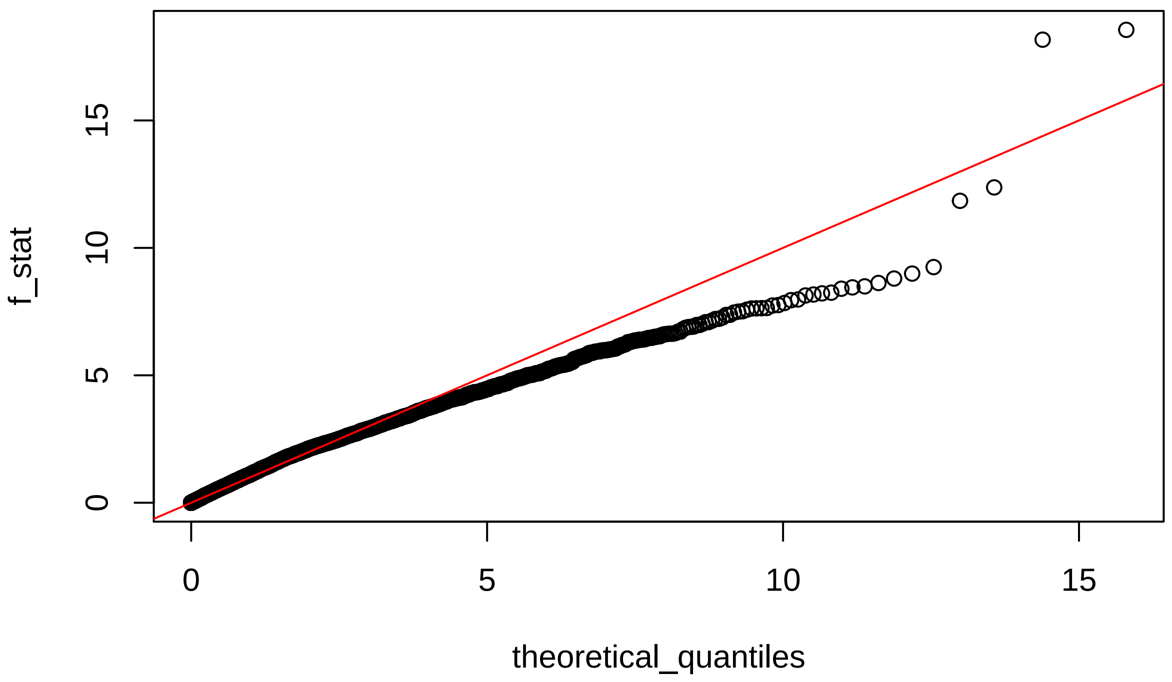


```
## [1] 0.6893
```

As expected, the t statistics for the coefficient β_2 deviate from a t distribution, while those for β_1 which is equal to zero fit the null distribution well.

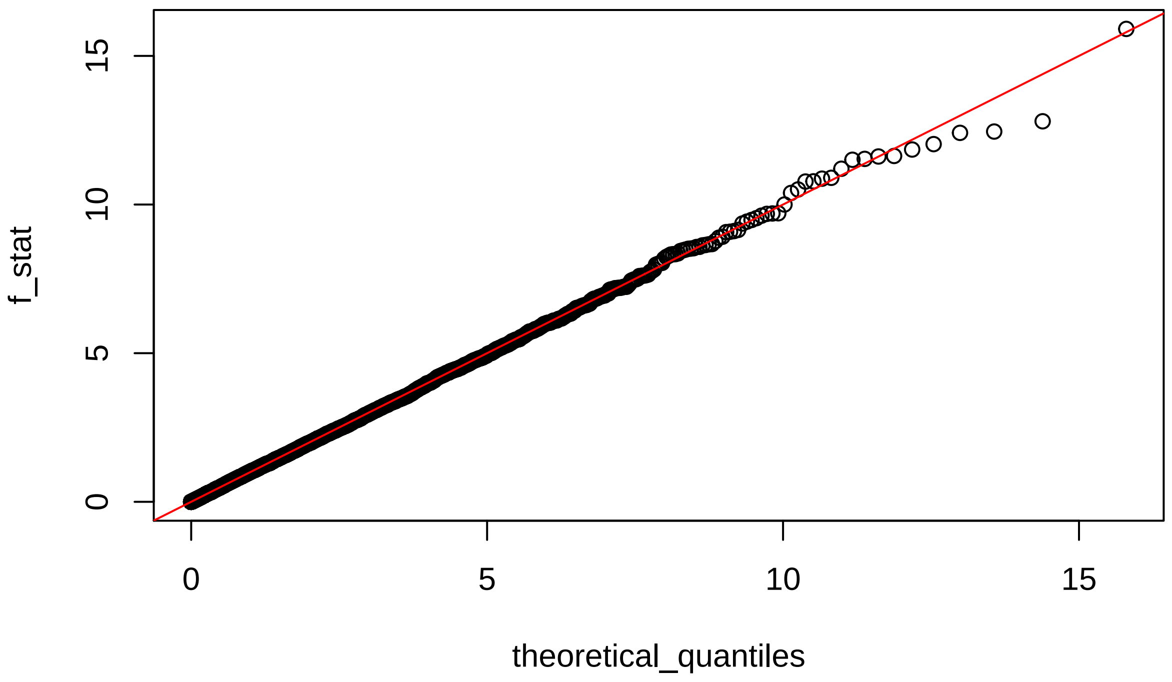
None of the results above are very surprising, since we can derive the exact distribution of the t statistics analytically. Now, we will study what happens when we change the error distribution — this is a form of *model mispecification.*

```
t_stat_mat_cauchy <- LinMod_sim(X,errors_gen = rcauchy)
f <- t_stat_mat_cauchy[1,]^2
qqplot_F(f,1,n-p)
```



```
## [1] 0.9586
```

```
t_stat_mat_exp <- LinMod_sim(X,errors_gen = function(x) rexp(x)-1)
f <- t_stat_mat_exp[1,]^2
qqplot_F(f,1,n-p)
```



```
## [1] 0.951
```

Interestingly, when we sample the errors from the shifted exponential, the t statistics seems to follow the t_{n-p} distribution. On the other hand, this distribution is a bad fit when the errors are Cauchy. Under the null hypothesis with Cauchy errors, the F statistic tends to fall below the values expected in the normal model. This suggests that using the t-test at level α would lead to a rate of Type-I errors lower than α; i.e. it is *safe* to use this test from the point of view of false rejections. In fact, the numerical output of `qqplot_F` suggests that the Type-I error for a test with level 5% is about 4.1%.

Of course, this observation is only valid for the design matrix we used in the simulation. Nonetheless, this type of simulation can suggest interesting research problems. For example, could this have something to do with the fact that the Cauchy distribution has heavy tails?

Fluctuation of the order statistics around expected values

You might have noticed that extreme order statistics seem to deviate more from their expected values in QQ plots. This has a clear explanation.

Recall that the ith order statistic $t_{(i)}$ of i.i.d. variables $t_1, t_2, \ldots, t_k$ has the same distribution as $G(U_{(i)})$, where G is the inverse CDF of t_1 and $U_{(i)}$ is the ith order statistic of k i.i.d. Uniform$(0,1)$ random variables. Now, $U_{(i)}$ has a Beta$(i, k+1-i)$ distribution, which has fluctuations of at most order $O(k^{-1})$ around its mean $i/(k+1)$. For large values of k we expect $U_{(i)}$ to be close to $i/(k+1)$ with high probability. The fluctuations of $F^{-1}(U_{(i)})$, then, will depend largely on the value of the gradient $|G'(i/(k+1))|$, which is typically largest when i is close to 1 or k (for variables ranging in $\mathbb{R}$) or close to k for strictly positive variables.

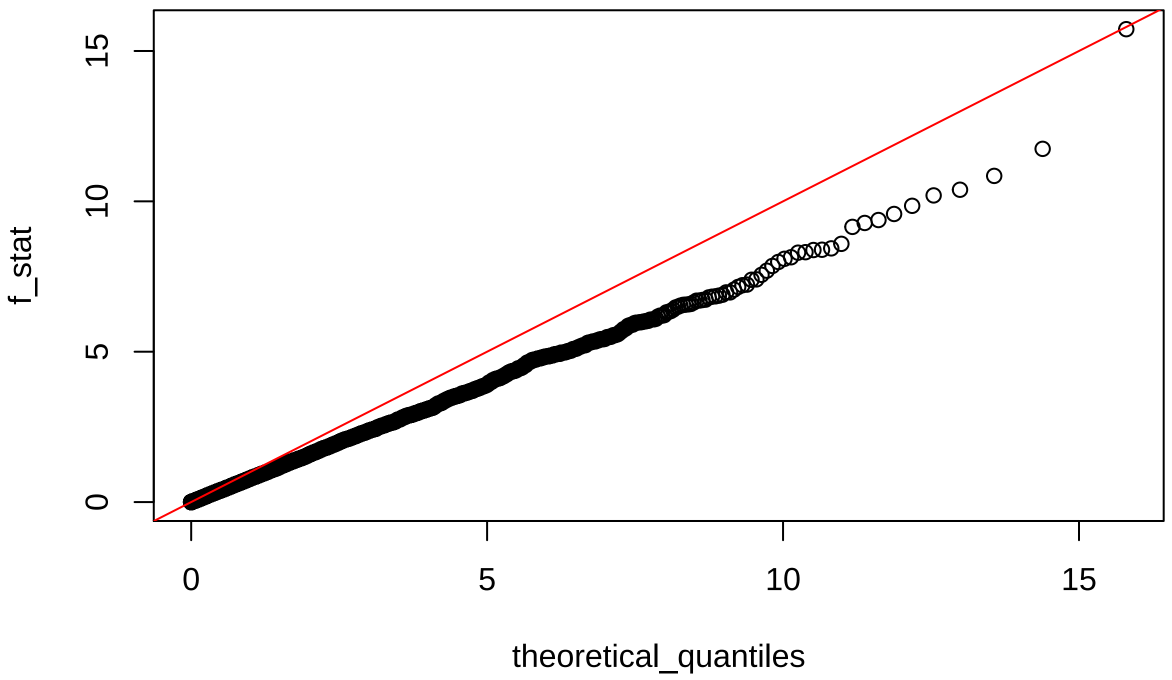
Exercise 1: Simulation with heteroskedasticity

We rewrite the function `LinMod_sim` as follows.

```r
HS_LinMod_sim <- function(X, Beta = rep(0, p), errors_gen = rnorm, B = 10000) {
  n <- nrow(X)
  p <- ncol(X)
  y_mat <- as.vector(X %*% Beta) + matrix(errors_gen(n*B), n, B) * rnorm(n)
  inv_gram <- solve(t(X) %*% X)
  P <- X %*% inv_gram %*% t(X)
  Beta_hat_mat <- inv_gram %*% t(X) %*% y_mat
  resid_mat <- y_mat - P %*% y_mat
  sigma_tilde_vec <- colSums(resid_mat^2) / (n - p)
  t_stat_mat <- Beta_hat_mat / sqrt(diag(inv_gram))
  t_stat_mat <- t_stat_mat / rep(sqrt(sigma_tilde_vec), each = p)
  return(t_stat_mat)
}
```

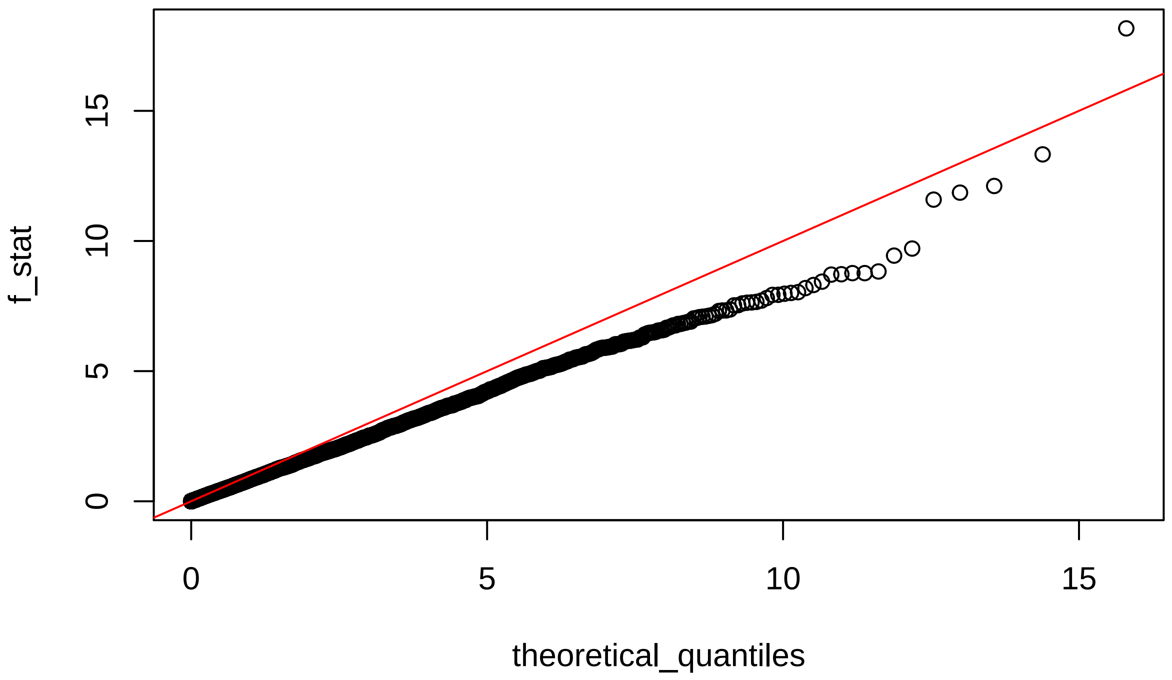
As before, we visualize the distribution of a t statistic.

```r
t_stat_mat_HS <- HS_LinMod_sim(X)
f <- t_stat_mat_HS[1,]^2
qqplot_F(f,1,n-p)
```



```
## [1] 0.9734
```

```
f <- t_stat_mat_HS[2,]^2
qqplot_F(f,1,n-p)
```



```
## [1] 0.9675
```

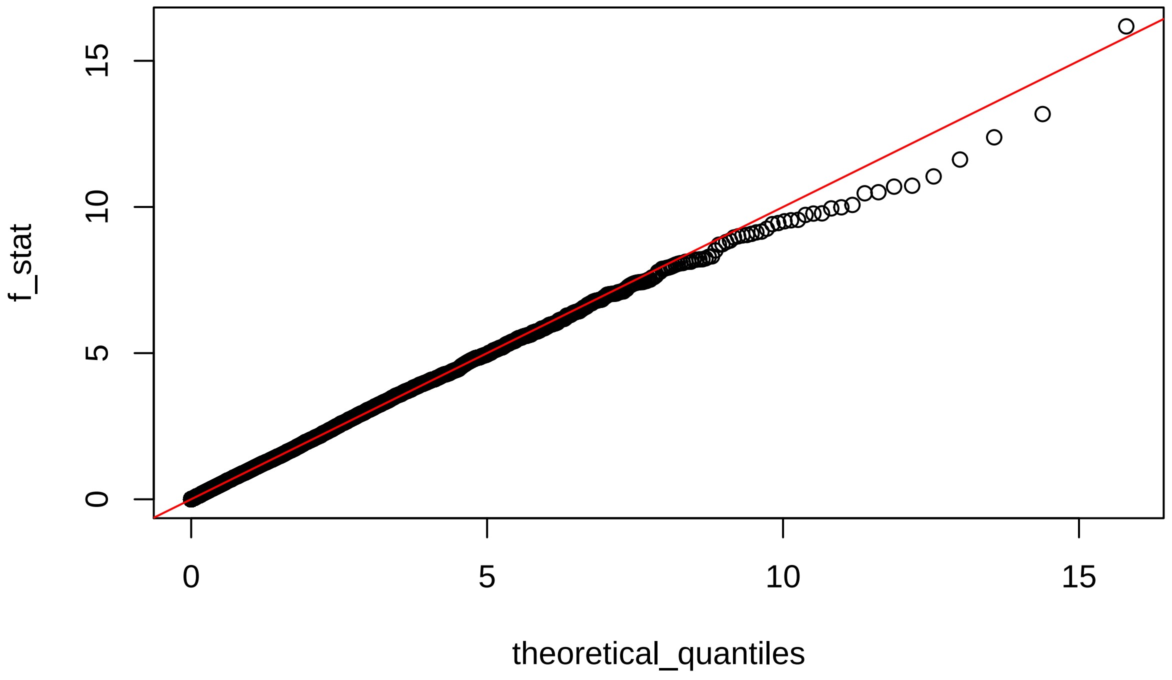
As expected, the null distribution is not t_{n-p}.

Let us now introduce partially known and simpler heteroskedasticity.

```
HSa_LinMod_sim <- function(X, a, Beta = rep(0, p), errors_gen = rnorm, B = 10000) {
  n <- nrow(X)
  p <- ncol(X)
  A <- diag(a^{-1})
  y_mat <- as.vector(X %*% Beta) + matrix(errors_gen(n*B), n, B) * rnorm(1) * sqrt(a)
  inv_mat <- solve(t(X) %*% A %*% X)
  Beta_hat_mat <- inv_mat %*% t(X) %*%  A %*% y_mat
  resid_mat <- y_mat - X %*% Beta_hat_mat
  sigma_tilde_vec <- colSums(resid_mat^2/a) / (n - p)
  t_stat_mat <- Beta_hat_mat / sqrt(diag(inv_mat))
  t_stat_mat <- t_stat_mat / rep(sqrt(sigma_tilde_vec), each = p)
  return(t_stat_mat)
}
```

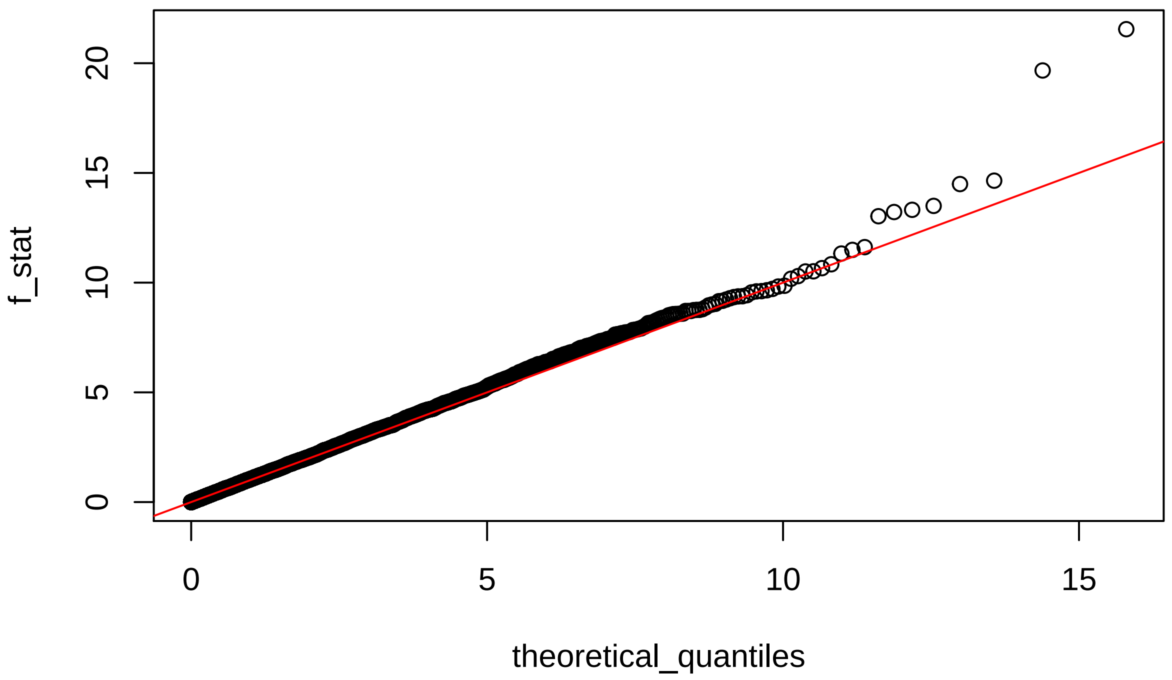
Note that the variations of this code with respect to `LinMod_sim` are justified by deriving the maximum likelihood estimators. Let us visualize the distribution of a t statistic.

```
t_stat_mat_HSa <- HSa_LinMod_sim(X,
  a = log(1+1:n))
f <- t_stat_mat_HSa[1,]^2
qqplot_F(f,1,n-p)
```



```
## [1] 0.9486
```

```
f <- t_stat_mat_HSa[2,]^2
qqplot_F(f,1,n-p)
```



```
## [1] 0.9437
```

We see that this time the null distribution seems to be much closer to t_{n-p}. This is because the underlying distribution generating the data belongs to a generalised linear model (see second half of the course), and the theory for these models justifies that, indeed, the null distribution is approximately t_{n-p}.

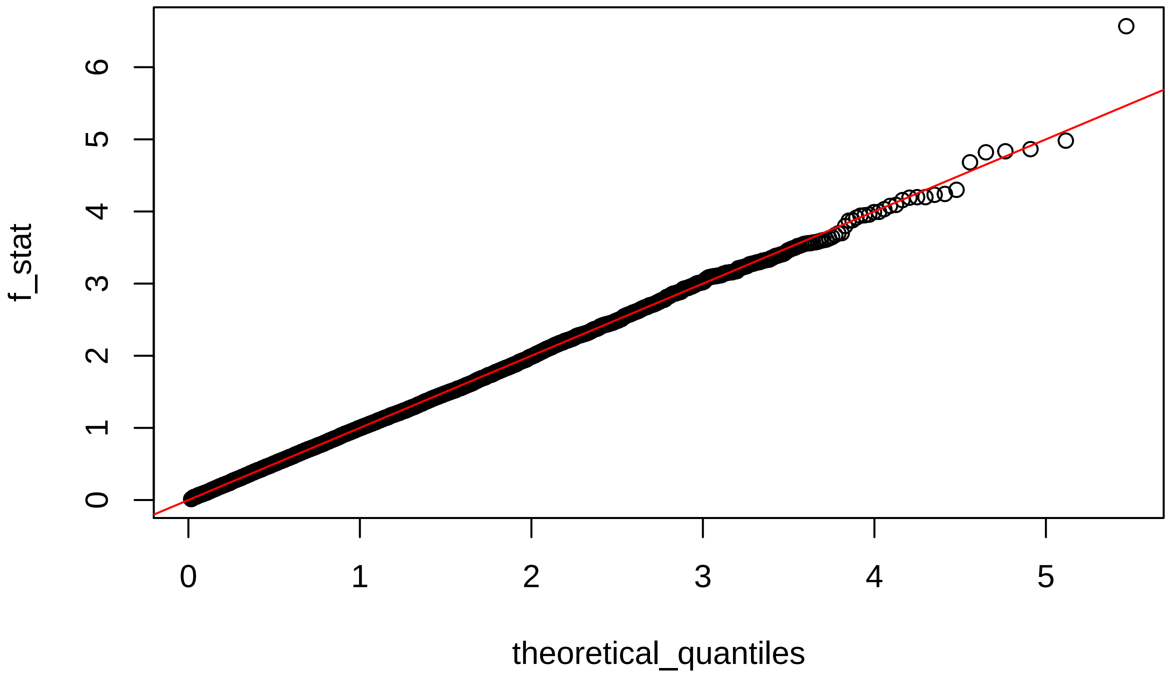
Exercise 2: Simulations of the F-test

```
LinMod_sim_F <- function(X, G0 = 1:ncol(X), Beta = rep(0, p), errors_gen = rnorm,
                         B = 10000) {
  n <- nrow(X)
  p <- ncol(X)
  p0 <- length(G0)
  X0 <- X[,G0]
  y_mat <- as.vector(X %*% Beta) + matrix(errors_gen(n*B), n, B)
  inv_gram <- solve(t(X) %*% X)
  P <- X %*% inv_gram %*% t(X)
  P0 <- X0 %*% solve(t(X0) %*% X0)  %*% t(X0)
  Beta_hat_mat <- inv_gram %*% t(X) %*% y_mat
  resid_mat <- y_mat - P %*% y_mat
  resid_mat0 <- (P-P0) %*% y_mat
  sigma_tilde_vec <- colSums(resid_mat^2) / (n - p)
  t_stat_mat <- Beta_hat_mat / sqrt(diag(inv_gram))
  t_stat_mat <- t_stat_mat / rep(sqrt(sigma_tilde_vec), each = p)
  F_vec <- colSums(resid_mat0^2) / (p-p0) / sigma_tilde_vec
  return(list("t"=t_stat_mat,"F"=F_vec))
}
```

The F statistic for the hypothesis $\beta_1 = 0$ has an $F_{p-p_0,n-p}$ distribution. We will verify this for a subset containing half of the predictors.

```
p0 = 5
out <- LinMod_sim_F(X,G0=1:p0)
```

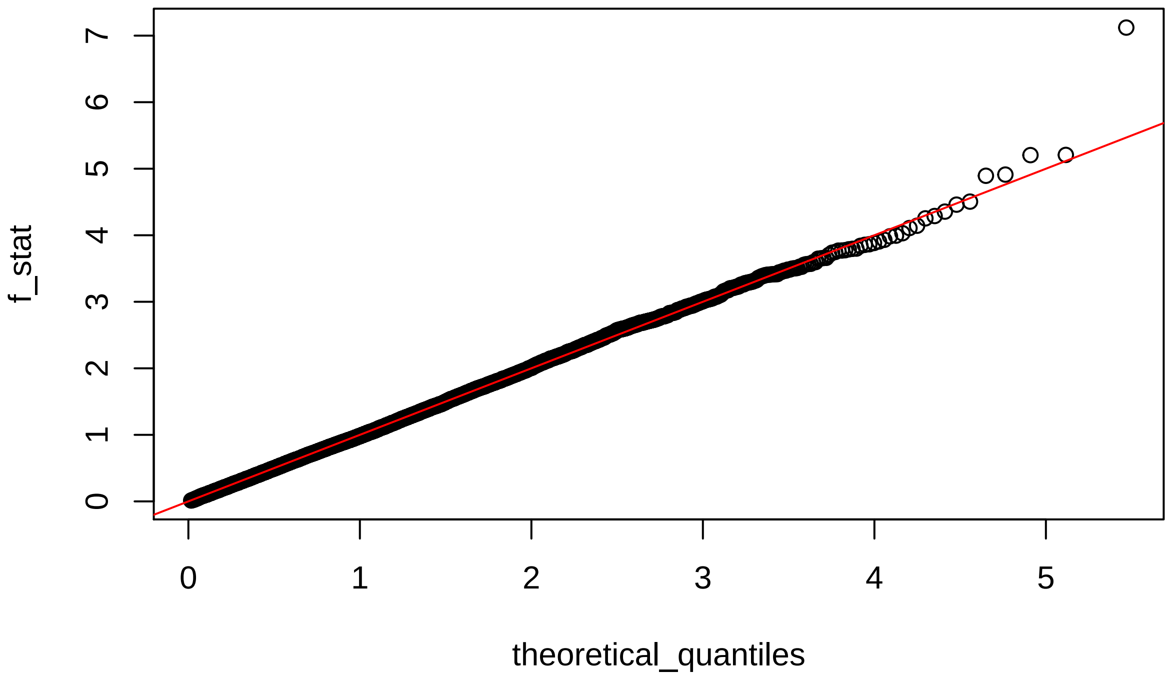
```
qqplot_F(out$F,p-p0,n-p)
```



```
## [1] 0.9496
```

The distribution of the F statistic doesn't change if we make any of the first p_0 coefficients different from zero.

```
out <- LinMod_sim_F(X,G0=1:p0,Beta=c(rnorm(p0),rep(0,p-p0)))
qqplot_F(out$F,p-p0,n-p)
```



```
## [1] 0.9478
```

Postscript

This file was produced with RMarkdown. You may view the source here in order to copy-paste code easily. You can also copy-paste it all and create an .Rmd file on RStutio, and regenerate the PDF output using the 'Knit' tab or modify the YAML header (or simply use the 'Knit' tab again) to produce an HTML version you can publish online. This is a fantastic way to share data analysis projects — and a great skill for any aspiring data bloggers among you — so I highly recommend that you take the practicals as an opportunity to learn it. However, note that creating these documents is not examinable, all that is examinable is understanding the solutions to the exercises.