1 Data and methodology

1.1 Data

Here comes text...

1.1.1 Descriptives

Table of summary statistics

Here comes a table and description of the stats

Table 1.1: Summary statistics of the returns

Statistics	Eurostoxx 50	Standardized Residuals
Minimum	-13.2404	-11.7732
Median	0.0357	-0.0193
Arithmetic Mean	0.0167	-0.0409
Maximum	10.4376	5.7126
Stdev	1.307	0.9992
Skewness	-0.31	-0.6327
	(0***)	(0***)
Excess Kurtosis	7.2083	5.134
	(0***)	(0***)
Jarque-Bera	19528.6196***	10431.0514***

Note: This table shows the descriptive statistics of the returns of over the period 1987-01-01 to 2

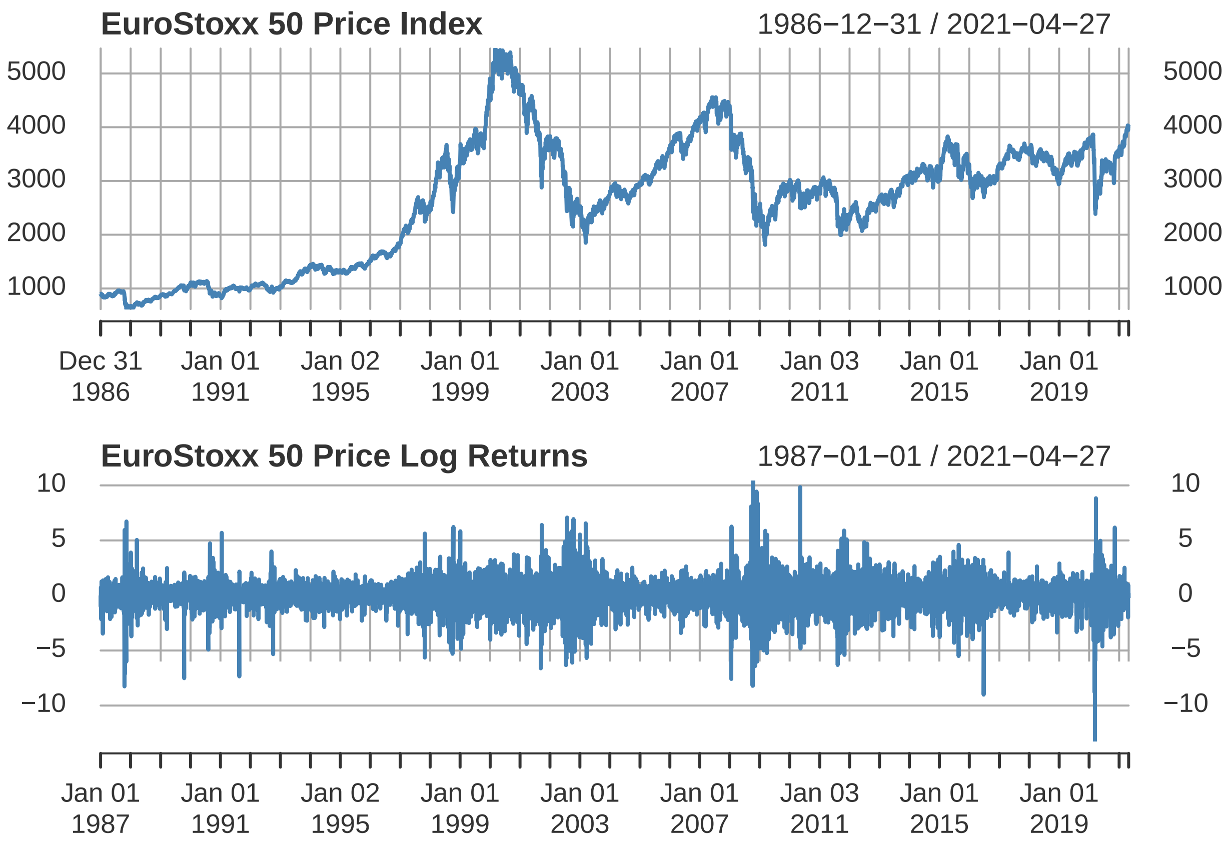
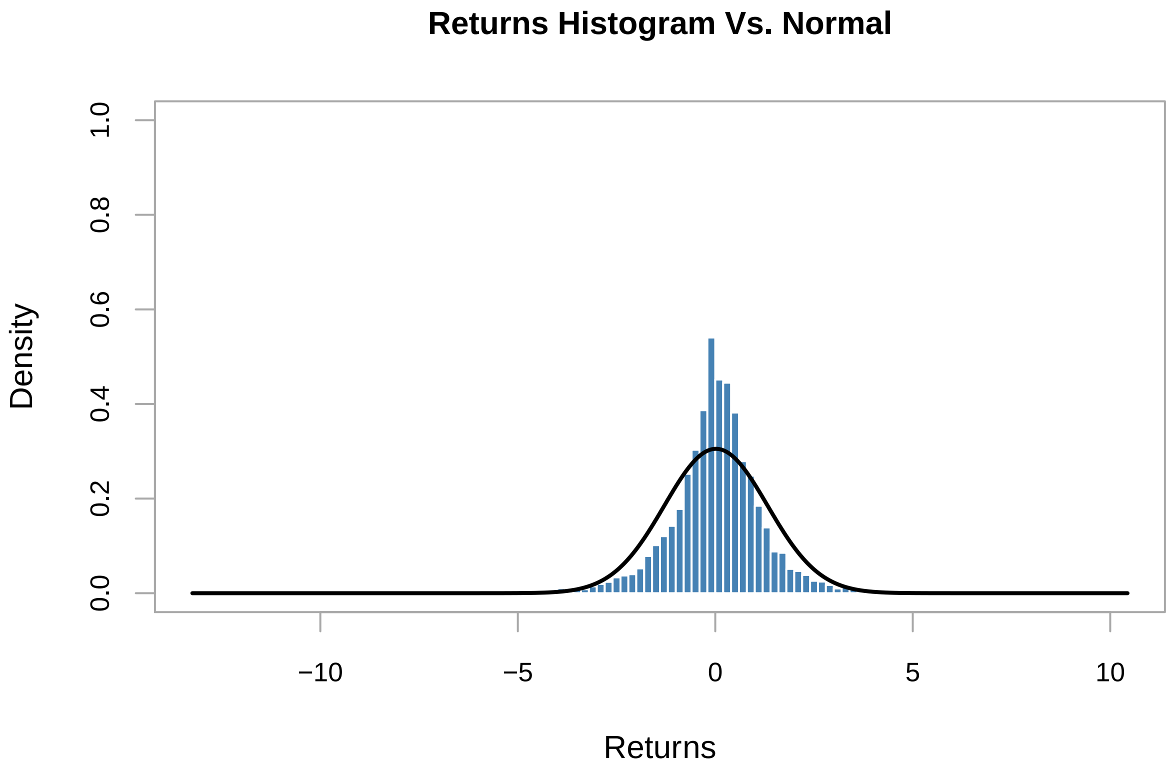


Figure 1.1: Eurostoxx 50 prices and returns

Descriptive figures



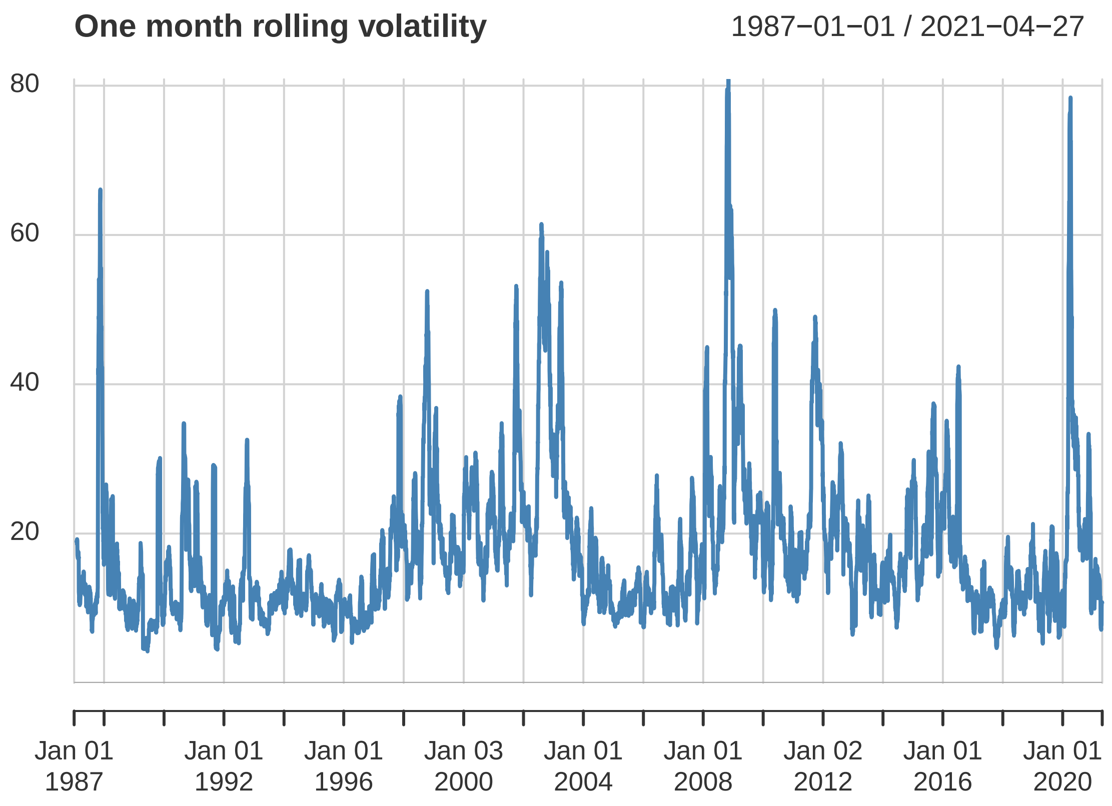


Figure 1.2: Eurostoxx 50 rolling volatility (22 days, calculated over 252 days)

As can be seen

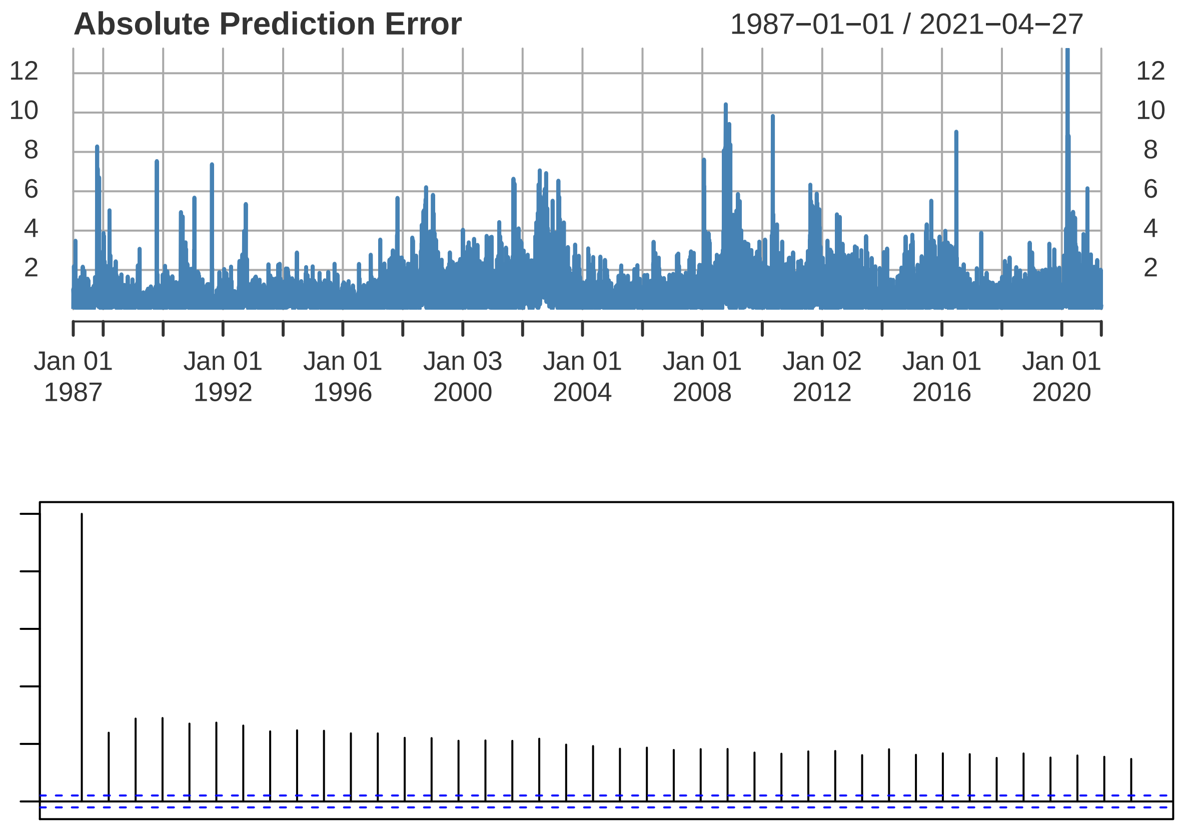


Figure 1.3: Absolute prediction errors

1.2 Methodology

As already mentioned in . . . , GARCH models GARCH, EGARCH, IGARCH, GJR-GARCH, NGARCH, TGARCH and NAGARCH (or TSGARCH) will be estimated. Additionally the distributions will be examined as well, including the normal, student-t distribution, skewed student-t distribution, generalized error distribution, skewed generalized error distribution and the skewed generalized t distribution.

They will be estimated using maximum likelihood. As already mentioned, fortunately, Alexios Ghalanos [1] has made it easy for us to implement this methodology in the R l anguage (version 3.6.1) with the package “rugarch” version 1.4-4 (R univariate garch), which gives us a bit more time to focus on the results and the interpretation. Additionally

Maximum likelihood estimation is a method to find the distribution parameters that best fit the observed data, through maximization of the likelihood function, or the computationally more efficient log-likelihood function (by taking the natural logarithm). It is assumed that the return data is i.i.d. and that there is some underlying parametrized density function f with one or more parameters θ that generates the data ((1.1)). These functions are based on the joint probability distribution of the observed data (equation (1.2)). Subsequently, the (log)likelihood function is maximized using an optimization algorithm (equation (1.3)).

$$y_1, y_2, ..., y_N \sim i.i.d y_i \sim f(y|\theta) \tag{1.1}$$

$$L(\theta) = \prod_{i=1}^{N} f(y_i|\theta) \log(L(\theta)) = \sum_{i=1}^{N} \log f(y_i|\theta) \tag{1.2}$$

$$\theta^* = arg \max_{\theta}[L]\theta^* = arg \max_{\theta}[\log(L)] \tag{1.3}$$

```r
# n_samples <- 25; true_rate <- 1; set.seed(1)
# exp_samples <- rexp(n = n_samples,
#                     rate = true_rate)
#
# sample_data <- exp_samples
# rate_fit_R <- fitdistrplus::fitdist(data = sample_data,
#                        distr = 'exp',
#                        method = 'mle')
#
# rate_fit_R$estimate
```

```r
# Normal
normfit_R <- fitdistrplus::fitdist(data = as.vector(R),
                     distr = "norm",
                     method = "mle")
normfit_R$estimate #estimated parameters
```

```
##       mean         sd
## 0.01668214 1.30689172
```

```
normfit_R$sd #estimated standard errors
```

```
##       mean         sd
## 0.01381119 0.00976596
```

```
normfit_R$loglik #max log-likelihood
```

```
## [1] -15101.73
```

```
#? # LR statistic


# T-dist
tstart <- list(df=1, ncp = 1)


tfit_R <- fitdistrplus::fitdist(data = as.vector(R),
                      distr = "t",
                      method = "mle", start = tstart)
tfit_R$estimate #estimated parameters
```

```
##         df        ncp
## 4.31096001 0.03168827
```

```
tfit_R$sd #estimated standard errors
```

```
##         df        ncp
## 0.14857777 0.01100453
```

```
tfit_R$loglik #max log-likelihood
```

```
## [1] -14149.5
```

```
#? # LR statistic


# GED distribution
gedstart <- list(mean=1, sd=1, nu=1)
```

```r
gedfit_R <- fitdistrplus::fitdist(data = as.vector(R),
                      distr = "ged",
                      method = "mle", start = gedstart)
gedfit_R$estimate
```

```
##       mean         sd         nu 
## 0.03160393 1.27550013 0.91274249
```

```r
gedfit_R$sd
```

```
##        mean          sd          nu 
## 0.008555584 0.015772159 0.016622605
```

```r
gedfit_R$loglik
```

```
## [1] -14009.53
```

```r
#? # LR statistic




# ST distribution
## Stephane's skewed student t is different from the rugarch package, so I would
## skewtstart <- list(xi=0, omega=2, alpha=2, nu=8)
## skewtfit_R <- fitdistrplus::fitdist(data = as.vector(R),
##                         distr = "st",
##                         method = "mle", skewtstart)
## skewtfit_R$estimate
## skewtfit_R$sd
## skewtfit_R$loglik


STstart <- list(mean=0,sd=2, nu = 8, xi=2)
STfit_R <- fitdistrplus::fitdist(data = as.vector(coredata(R)), distr = "sstd", met
STfit_R$estimate
```

```
##      mean        sd        nu        xi
## 0.0187729 1.4868913 2.7847974 0.9485825
```

```
STfit_R$sd
```

```
##       mean         sd         nu         xi
## 0.01375064 0.05550991 0.09972285 0.01270650
```

```
STfit_R$loglik
```

```
## [1] -13997.35
```

```
## ? # LR statistic


# ST <- rugarch::fitdist("sstd", x = R)
# ST$pars # parameters are very similar to the ones we got with fitdistrplus, whi


#SGT distribution
X.f = X ~ coredata(R)
start = list(mu=0,sigma=2, lambda = 0, p=2, q=8)#list(mu=0,sigma=2, lambda = 0.5,
result = sgt.mle(X.f = X.f, start = start)
summary(result)
```

```
## Skewed Generalized T MLE Fit
## Best Result with BFGS Maximization
## Convergence Code 0: Successful Convergence
## Iterations: NA, Log-Likelihood: -13973.01
##
##           Est. Std. Err.       z  P>|z|
## mu      0.0204    0.0131  1.5574 0.1194
## sigma   1.3214    0.0261 50.5971 0.0000 ***
## lambda -0.0397    0.0126 -3.1583 0.0016  **
## p       1.3818    0.0708 19.5077 0.0000 ***
## q       3.3093    0.5333  6.2058 0.0000 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# SGTstart <- list(mu=0,sigma=2, lambda = 0.5, p=2, q=8)
# SGTfit_R <- fitdistrplus::fitdist(data = as.vector(coredata(R)), distr = "sgt",
# summary(SGTfit_R)
## ? # LR statistic
```

1.2.1 Control Tests

Unconditional coverage test of Kupiec [2]

A number of tests are computed to see if the value-at-risk estimations capture the actual losses well. A first one is the unconditional coverage test by Kupiec [2]. The unconditional coverage or proportion of failures method tests if the actual value-at-risk exceedances are consistent with the expected exceedances (a chosen percentile, e.g. 1% percentile) of the VaR model. Following Kupiec [2] and Ghalanos [3], the number of exceedence follow a binomial distribution (with thus probability equal to the significance level or expected proportion) under the null hypothesis of a correct VaR model. The test is conducted as a likelihood ratio test with statistic like in equation (1.4), with p the probability of an exceedence for a confidence level, N the sample size and X the number of exceedence. The null hypothesis states that the test statistic LR^{uc} is χ^2-distributed with one degree of freedom or that the probability of failure $\hat{p}$ is equal to the chosen percentile α.

$$LR^{uc} = -2\ln\left(\frac{(1-p)^{N-X}p^X}{\left(1-\frac{X}{N}\right)^{N-X}\left(\frac{X}{N}\right)^X}\right) \tag{1.4}$$

Conditional coverage test of Christoffersen, Hahn, and Inoue [4]

Christoffersen, Hahn, and Inoue [4] proposed the conditional coverage test. It is tests for unconditional covrage and serial independence. The serial independence is important while the LR^{uc} can give a false picture while at any point in time it classifies inaccurate VaR estimates as “acceptably accurate” [5]. For a certain VaR estimate an indicator variable, $I_t(\alpha)$, is computed as equation (1.5).

$$I_t(\alpha) = \begin{cases} 1 & \text{if exceedence occurs} \\ 0 & \text{if no exceedence occurs} \end{cases}. \tag{1.5}$$

It involves a likelihood ratio test’s null hypothesis is that the statistic is χ^2-distributed with two degrees of freedom or that the probability of violation $\hat{p}$ (unconditional coverage) as well as the conditional coverage (independence) is equal to the chosen percentile α.

Dynamic quantile test

.@engle2004 with the aim to provide completeness to the conditional coverage test of Christoffersen, Hahn, and Inoue [4] developed the Dynamic quantile test. It consists in testing some restriction in a

References

[1] Alexios Ghalanos. *rugarch: Univariate GARCH models.* R package version 1.4-4. 2020.

[2] P.H. Kupiec. “Techniques for Verifying the Accuracy of Risk Measurement Models”. In: *Journal of Derivatives* 3.2 (1995), pp. 73–84.

[3] Alexios Ghalanos. *Introduction to the rugarch package. (Version 1.4-3).* Tech. rep. 2020. URL: `http://cran.r-project.org/web/packages/`.

[4] Peter Christoffersen, Jinyong Hahn, and Atsushi Inoue. “Testing and comparing Value-at-Risk measures”. In: *Journal of Empirical Finance* 8.3 (July 2001), pp. 325–342. URL: `https://linkinghub.elsevier.com/retrieve/pii/S0927539801000251`.

[5] Turan G. Bali and Panayiotis Theodossiou. “A conditional-SGT-VaR approach with alternative GARCH models”. In: *Annals of Operations Research* 151.1 (Feb. 22, 2007), pp. 241–267. URL: `http://link.springer.com/10.1007/s10479-006-0118-4`.