LUISS Guido Carli
LIBERA UNIVERSITÀ INTERNAZIONALE DEGLI STUDI SOCIALI

Dipartimento di Economia e Finanza

Cattedra di Econometria per la Finanza

Modelli GARCH per la costruzione di un portafoglio di criptovalute e la stima del value at risk

RELATORE
Prof. Stefano Grassi

CANDIDATO
David Jesai Vitagliano
Matr. 701561

CORRELATORE
Prof. Santucci De Magistris Paolo

ANNO ACCADEMICO 2018/2019

Indice

Indice delle Figure

Indice delle Tabelle

INTRODUZIONE

Modellare la volatilità è diventato molto importante per l'attività di risk management. Infatti, dopo la crisi finanziaria globale del 2008, il quadro normativo internazionale di Basilea III ha imposto dei requisiti patrimoniali più rigorosi e sono stati sviluppati sistemi rafforzati di gestione del rischio. Da allora il sistema finanziario internazionale ha dovuto affrontare una nuova sfida, vale a dire l'introduzione di criptovalute decentrate, il primo è Bitcoin, che è stato creato nel 2009. A differenza delle valute tradizionali, le criptovalute si basano sulla prova crittografica, che fornisce molti vantaggi rispetto ai metodi di pagamento tradizionali (come le carte di credito) tra cui alta liquidità, minori costi di transazione, e l'anonimato. L'interesse per Bitcoin e altre criptovalute è aumentato notevolmente negli ultimi anni. La loro capitalizzazione di mercato è aumentata da circa 18 miliardi di dollari americani all'inizio del 2017 a quasi 600 miliardi alla fine di quell'anno, e gli alti rendimenti hanno attirato nuovi investitori. Inoltre, due grandi borse, ad es. il Chicago Mercantile Exchange (CME) e il Chicago Board Options Exchange (CBOE), hanno iniziato a commerciare futures su Bitcoin. A seguito di questi sviluppi, le banche centrali hanno affrontato la questione se le criptovalute debbano o meno essere regolamentate, date le numerose questioni tecniche e giuridiche coinvolte. La letteratura sulle criptovalute è stata inizialmente dominata da studi sulla sicurezza, aspetti etici e legali di Bitcoin. Invece, la letteratura recente ha esaminato le criptovalute da un punto di vista economico. Tuttavia, poco è noto circa il comportamento dei rendimenti e della volatilità delle criptovalute. Un aspetto particolarmente interessante è se i prezzi altamente volatili delle criptovalute evolvono in modo casuale nel tempo o mostrano una certa prevedibilità.

Come viene ricordato in Zouheir Mighri et al. (2019), ci sono diversi risultati che sono stati ottenuti in diversi studi che riporto qui di seguito. Come sostenuto da Cheah e Fry (2015), se il bitcoin fosse una vera unità di conto, o una forma di riserva di valore, non mostrerebbe tale volatilità espressa da bolle e crash. Secondo Dwyer (2015), la media della volatilità mensile del bitcoin è superiore a quella dell'oro o di un insieme di valute estere. Inoltre, trova che la più bassa volatilità mensile per il bitcoin è inferiore alla più alta volatilità mensile per oro e valute. Inoltre, Brière et al. (2015) mostrano che il bitcoin offre significativi benefici di diversificazione per gli investitori, mentre Urquhart (2016) mostra che i rendimenti di bitcoin non seguono una *random walk*. Utilizzando un modello asimmetrico GARCH, Dyhrberg (2016a) dimostra che il bitcoin può essere utile nella gestione del rischio e ideale per gli investitori avversi al rischio in previsione di shock negativi al mercato. Inoltre, Dyhrberg (2016 a, b) mostra che il Bitcoin ha capacità di copertura simili a quelle dell'oro e del dollaro, e come tali può essere impiegato per la gestione del rischio. Inoltre, Balcilar et al. (2017) mostrano che il volume di scambi del bitcoin può prevedere i rendimenti tranne nei regimi di mercato *bull* e *bear* e che il volume non può prevedere la volatilità dei rendimenti bitcoin. Questa tesi ha lo scopo di analizzare la volatilità di quattro criptovalute (Bitcoin, Ethereum, Litecoin e Monero) utilizzando diversi modelli GARCH.

In primo luogo, modellizziamo la volatilità del bitcoin con diversi GARCH univariati e con tre diverse distribuzioni. I GARCH univariati utilizzati sono: SGARCH, EGARCH, IGARCH, TGARCH, GJR-GARCH. I modelli asimmetrici (EGARCH, TGARCH, GJR-GARCH) sono utilizzati appunto per tentare di catturare eventuali effetti di leverage che di solito sono presenti nelle serie temporali finanziarie. Infatti, tali modelli nascono proprio perché ci si rende conto che le reazioni degli investitori sono diverse a seconda che ci siano shock negativi o positivi, generando effetti diversi sulla volatilità dei rendimenti. Generalmente, per le serie di rendimenti finanziari succede che in caso di shock negativi dovuto a "bad news", cioè quando i valori dei rendimenti sono al di sotto delle aspettative, l'impatto sulla volatilità è più grande e quindi quest'ultima registra dei valori più elevati rispetto a quelli che osserviamo in caso di shock positivi (dovuto a "good news") della stessa portata. Per tenere conto dell'eccesso di curtosi (leptocurtosi) e di skewness, che di solito è presente nelle serie finanziarie, utilizziamo tre diverse distribuzioni: t di student, *generalized error distribution (*GED) e normale inversa gaussiana (NIG). Inoltre, per la scelta dei modelli vengono considerati anche i criteri di Akaike, Bayes, Shibata e Hannan-Quinn. Dopo una prima analisi univariata del Bitcoin, si passa al multivariato andando a stimare la matrice di covarianza/varianza condizionale e la matrice di correlazione condizionale delle quattro criptovalute citate in precedenza. Tale analisi viene effettuata utilizzando il modello con correlazioni condizionali dinamiche (DCC). Tale modello viene proposto in vari studi quali Christodoulakis e Satchell (2002), Tse e Tsui (2002) e Engle (2002). Il vantaggio principale dei modelli DCC è che, essendo stimati in due fasi, il processo di stima è meno complesso nel caso in cui N sia alto rispetto ad altri modelli GARCH multivariati. La prima fase consiste nella stima di diversi GARCH univariati, mentre nella seconda fase si va a stimare la matrice di correlazione. Una volta concluso il processo di stima del DCC-GARCH, costruiremo il nostro portafoglio di criptovalute. Il rendimento atteso e il rischio misurato dalla varianza (o dalla standard deviation, ovvero la radice quadrata della varianza) sono le due caratteristiche principali di un portafoglio di investimento. Pertanto, utilizzeremo la frontiera efficiente, introdotta dal premio Nobel per l'economia Harry Markowitz nel 1952, per indicare il portafoglio ottimale. La frontiera efficiente è l'insieme dei portafogli ottimali che offrono il rendimento atteso più elevato per un determinato livello di rischio o il rischio più basso per un determinato livello di rendimento atteso. La frontiera efficiente verrà costruita sulla base dei rendimenti attesi e delle varianze e covarianze attese stimate dal modello DCC-GARCH. Con i pesi ottimali ottenuti stimiamo il value at risk della serie dei rendimenti del portafoglio con due metodologie: historical simulation e attraverso la stima di modelli GARCH univariati. Inoltre, per verificare la bontà di tali stime utilizziamo dei test statistici: violation ratio, test di Kupiec e test di Christoffersen. Il primo capitolo sarà dedicato alla letteratura dei GARCH univariati e multivariati. Nel secondo capitolo mostreremo l'analisi univariata, per quanto riguarda il Bitcoin, e multivariata delle criptovalute con l'utilizzo del DCC-GARCH. Nel terzo capitolo costruiremo un portafoglio di criptovalute, calcolandone la frontiera efficiente. Inoltre, con i pesi ottimali ottenuti stimeremo il value at risk con le metodologie di cui sopra e ne verificheremo la bontà di tali stime utilizzando test statistici.

1 Fatti stilizzati delle serie di rendimenti finanziari

Molti studi sulle proprietà delle serie di rendimenti finanziari hanno dimostrato che i rendimenti esibiscono tre proprietà statistiche che sono presenti in molte, se non in tutte, le serie finanziarie. Tali proprietà sono spesso chiamate come i tre fatti stilizzati delle serie di rendimenti finanziari (Danielsson, 2011):

- *Volatility clusters*
- *Code grasse*
- *Dipendenza non lineare*

La prima proprietà si basa sull'osservazione che grandi cambiamenti o, allo stesso modo, piccoli cambiamenti tendono a raggrupparsi tra di loro, così capita spesso di osservare molti giorni di alta volatilità seguiti da molti giorni di bassa volatilità.

La seconda proprietà è, invece, basata sul fatto che le serie finanziarie occasionalmente presentano dei rendimenti positivi o negativi molto elevati e questo fa sì che le code siano più grasse di una normale avendo valori estremi elevati.

Infine, la dipendenza non lineare è da attribuire a come i rendimenti multivariati si relazionano tra di loro. Se i rendimenti sono linearmente dipendenti, la correlazione descrive come si muovono insieme. Se invece sono caratterizzati da una dipendenza non lineare, la correlazione tra i diversi rendimenti dipende dalla grandezza dei risultati. Ad esempio, si è spesso osservato che le correlazioni sono più basse in *bull markets* che in *bear markets*, mentre durante crisi finanziare tende a raggiungere il 100% (Danielsson, 2011).

1.1 La volatilità

Gli economisti finanziari sono sempre alla ricerca di nuovi modelli per modellizzare la volatilità nei rendimenti degli assets. Infatti, la volatilità è un elemento importante in quanto misura del rischio di un asset e, quindi, se la volatilità è alta gli investitori chiederanno un premio più alto per investire in tali assets assumendosi un rischio maggiore. Guardando la figura 1 (Danielsson, 2011) che mostra la volatilità annuale dall'anno 1980, vediamo che il periodo con la più alta volatilità è durante la recente crisi finanziaria, 2007-2009, seguito dal crollo del mercato azionario del 1987. Il periodo più calmo è l'anno 1995 e tra il 2004-2006. Tuttavia, il fatto che la volatilità fosse bassa non implica che il rischio nei mercati finanziari in quel periodo fosse basso, in quanto la volatilità può essere bassa mentre le code sono grasse. In altre parole, è possibile che una serie finanziaria abbia bassa volatilità ma presenti dei valori molto più estremi di un'altra serie con alta volatilità. Questo è il motivo per il quale la volatilità è una ingannevole misura del rischio. Pertanto, banche e istituzioni finanziare applicano i c.d. modelli "value-at-risk" per valutare i rischi che stanno assumendo nello svolgimento delle loro attività. Dopo la crisi finanziaria del 2007-2009, le regole riguardo la valutazione dei rischi sono diventate sempre più stringenti per Banche e grandi istituti finanziari che hanno dovuto

implementare modelli sempre più accurati. Quindi, si può ben capire come la modellizzazione e la previsione della volatilità o, in altre parole, la struttura della covarianza dei rendimenti degli asset sia di primaria importanza per tali soggetti. Originariamente, i rendimenti venivano modellati come indipendenti e identicamente distribuiti nel tempo. Tuttavia, le serie di rendimenti di assets finanziari osservate settimanalmente o a maggiore frequenza non sono affatto indipendenti. Infatti, queste serie anche se sono incorrelate o quasi incorrelate tra di loro, presentano della dipendenza. Modelli come Autoregressive Conditional Heteroskedasticity (ARCH) e la sua forma generalizzata (GARCH) sono i modi più popolari per parametrizzare questa dipendenza. Utilizzando questi modelli andremo a specificare media e varianza di alcune criptovalute al fine di creare un portafoglio.

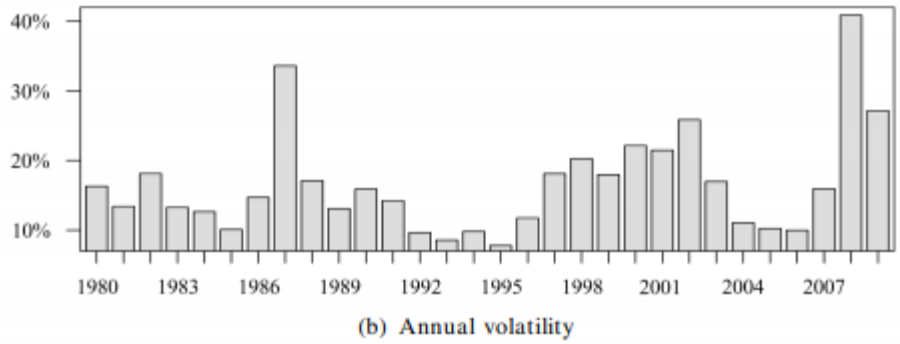

Figura 1: Volatilità annuale (1980-2009). Fonte: Danielsson, J., Financial Risk Forecasting., p. 10. WILEY, Londra (2011).

1.2 Modelli ARCH

Consideriamo una variabile casuale ε_t con una media e una varianza condizionata ad un set informativo I_{t-1}. Il modello ARCH di ε_t ha le seguenti proprietà:

$$E(\varepsilon_t|I_{t-1}) = 0 \qquad (1)$$

e ciò implica che tale processo è serialmente incorrelato condizionatamente allo stesso set informativo[1]. La varianza condizionale $h_t = E(\varepsilon_t^2|I_{t-1})$ è una funzione parametrica non banale con valori positivi di I_{t-1}.
Inoltre, ε_t può essere scorporato in due parti:

$$\varepsilon_t = z_t h_t^{1/2} \qquad (2)$$

dove z_t è una sequenza di variabili i.i.d. con media zero e varianza uno. Da queste premesse si ha che $\varepsilon_t|I_{t-1} \sim D(0, h_t)$, dove D indica distribuzione (tipicamente si assume abbia distribuzione normale). La seguente varianza condizionale definisce un modello ARCH di ordine q:

$$h_t = w + \sum_{j=1}^{q} \alpha_j \varepsilon_{t-j}^2 \qquad (3)$$

[1] Avere informazioni sui valori passati di ε_t non mi dice nulla su quali saranno i futuri valori di tale variabile.

dove w ≥ 0, $\alpha_j \geq 0$, per j = 1,2,…,q sono i parametri da stimare nel modello. Quindi la varianza condizionale viene specificata come una funzione lineare dei quadrati dei valori passati di ε_t. Inoltre, tale modello può essere riscritto facendo uso dell'operatore ritardo[2]:

$$h_t = w + A(L)\varepsilon_t^2 \qquad (4)$$

dove $A(L) = \alpha_1 L, \alpha_2 L^2 + \cdots + \alpha_q L^q$ è il polinomio dell'operatore ritardo. L'ARCH(q) può essere definito stazionario[3] in covarianza quando le radici del polinomio 1-A(L) sono maggiori di uno in modulo. Per rispettare tale condizione quindi:

$$\sum_{j=1}^{q} \alpha_j < 1 \qquad (5)$$

Se il processo è stazionario $E(\varepsilon_t^2) = \sigma^2 < \infty$, ossia la varianza non condizionale è finita ed assume il seguente valore:

$$Var(\varepsilon_t) = \frac{w}{1-\sum_{j=1}^{q} \alpha_j} \qquad (6)$$

1.3 Modello GARCH

Nella pratica, il modello ARCH è stato sostituito dalla sua forma generalizzata (GARCH) proposto sia da Bollerslev (1986) che da Taylor (1986) indipendentemente l'uno dall'altro. In questo modello, a differenza dell'ARCH, la varianza condizionale è funzione lineare anche dei suoi stessi *lags* ed è definito nel seguente modo:

$$h_t = w + \sum_{j=1}^{q} \alpha_j \varepsilon_{t-j}^2 + \sum_{j=1}^{q} \beta_j h_{t-j} \qquad (7)$$

La condizione necessaria e sufficiente per cui sia garantita la positività della varianza condizionale è w ≥ 0, $\alpha_j \geq 0,\ j = 1, \dots, q;\ \beta_j \geq 0,\ j = 1, \dots, p$. Facendo ancora uso dell'operatore ritardo è possibile riscriverlo come:

$$h_t = w + A(L)\varepsilon_t^2 + B(L)h_t \qquad (8)$$

Bollerslev (1986) mostrò come il modello sia stazionario in covarianza quando $\sum_{j=1}^{q} \alpha_j + \sum_{j=1}^{p} \beta_j < 1$, cioè quando le radici del polinomio $1 - A(L) - B(L)$ cadono al di fuori del cerchio unitario. Quando il processo è stazionario la varianza non condizionale assume il seguente valore:

$$Var(\varepsilon_t) = \frac{w}{1-\sum_{j=1}^{q} \alpha_j + \sum_{j=1}^{p} \beta_j} \qquad (9)$$

Consideriamo un GARCH (1, 1), definito come $h_t = w + \alpha\varepsilon_{t-1}^2 + \beta h_{t-1}$, se sostituiamo h_{t-1} otteniamo:

$$h_t = w + \alpha\varepsilon_{t-1}^2 + \beta(w + \alpha\varepsilon_{t-2}^2 + \beta h_{t-2})$$

oppure

$$h_t = w + \beta w + \alpha\varepsilon_{t-1}^2 + \alpha\beta\varepsilon_{t-2}^2 + \beta^2 h_{t-2}$$

---

[2] Date delle serie storiche $Y = \{Y_1, Y_{2,\dots}\}$, l'operatore ritardo funziona in modo tale che $L^k Y_t = Y_{t-k}$.

[3] cfr. Riccardo Jack Lucchetti, Appunti di analisi di serie storiche, pp 2-4 (2015). Disponibile su: http://www2.econ.univpm.it/servizi/hpp/lucchetti/didattica/matvario/procstoc.pdf

Sostituendo per h_{t-2} e iterando il processo, vediamo che il peso attribuito a ε_{t-j}^2 è $\alpha\beta^{i-1}$. I pesi quindi declinano esponenzialmente ad una velocità β. Quindi, β può essere considerato un tasso di decadimento che definisce importanza relativamente alle osservazioni di ε_t^2 nel calcolare la varianza corrente. Ad esempio, se $\beta = 0{,}9$, ε_{t-2}^2 è solo il 90% importante quanto ε_{t-1}^2; ε_{t-3}^2 è l'81% importante quanto ε_{t-1}^2; e così via. Inoltre, la varianza condizionale fluttua intorno la varianza non condizionale e tende ad essa. Questo fenomeno è definito come "*mean reversion"*.

1.4 Estensioni del modello GARCH: modelli con asimmetria e GARCH integrato

I modelli asimmetrici nascono perché ci si è resi conto che l'impatto sulla volatilità dei rendimenti è diverso a seconda che ci siano shock negativi o positivi. Infatti, in caso di shock negativi dovuto a "bad news", cioè quando i valori dei rendimenti sono al di sotto delle aspettative, l'impatto sulla volatilità è più grande e quindi quest'ultima registra dei valori più elevati rispetto a quelli che osserviamo in caso di shock positivi (dovuto a "good news") della stessa portata.

1.4.1 GJR GARCH

Il modello GJR GARCH di Glosten, Jagannathan e Runkle fa parte della famiglia dei modelli asimmetrici ed è definito nel seguente modo (consideriamo il caso (1,1) per semplicità):

$$h_t = w + (\alpha + \delta I_{t-1})\varepsilon_{t-1}^2 + \beta h_{t-1} \qquad (10)$$

dove I_{t-1} è una funzione indicatrice che vale uno se il rendimento al tempo *t-1* è negativo altrimenti vale zero. Infatti, quando i rendimenti sono positivi il coefficiente di ε_{t-1}^2 sarà α ed avremo semplicemente un modello GARCH (1, 1): $h_t = w + \alpha\varepsilon_{t-1}^2 + \beta h_{t-1}$. Mentre nel caso in cui i rendimenti sono negativi, il coefficiente di ε_{t-1}^2 sarà $(\alpha + \delta) > \alpha$ quindi tale parametro sarà in grado di catturare meglio le asimmetrie di cui abbiamo parlato in precedenza.

1.4.2 TGARCH

Il modello Threshold GARCH è uguale allla varianza condizionale viene sostituita dalla standard deviation condizionale, quindi al posto di h_t avremo $h_t^{1/2}$, h_{t-1} sarà rimpiazzato da $h_{t-1}^{1/2}$ e ε_{t-1}^2 da $|\varepsilon_{t-1}|$. In Zakoïan (1994), la standard deviation è definita come:

$$h_t^{1/2} = w + (\alpha + \delta I_{t-1})|\varepsilon_{t-1}| + \beta h_{t-1}^{1/2} \qquad (11)$$

1.4.3 EGARCH

Il modello EGARCH fu proposto da Nelson (1991). Nelson e Cao (1992) sostenevano che i vincoli di non negatività nel modello GARCH lineare erano troppo restrittivi. Il modello EGARCH è così definito:

$$ln\, h_t = w + \alpha g(z_{t-1}) + \beta\, ln\, h_{t-1} \qquad (12)$$

dove $g(z_{t-1}) = \theta z_{t-1} + \gamma[|z_{t-1}| - E|z_{t-1}|]$. Tale modello, a differenza del GARCH lineare, non ha bisogno di restrizioni per garantire la positività della varianza condizionale dato che il logaritmo rimuove i vincoli di non negatività sui parametri. L'asimmetria dipende dal parametro α. Ad esempio, quando $\alpha < 0$, $\ln h_t$ sarebbe più grande della sua media w se $z_{t-1} < 0$ e sarebbe più piccolo se $z_{t-1} > 0$. Questo significa che quando $\alpha < 0$, le notizie negative hanno un effetto maggiore rispetto alle notizie positive. Dall'altro lato, quando $\alpha > 0$, le notizie positive hanno un effetto più grande sulla varianza condizionale rispetto alle notizie negative.

1.4.4 GARCH integrato (IGARCH)

Nel 1986 Engle e Bollerslev proposero il modello GARCH integrato (IGARCH). Molti studi hanno mostrato che la somma dei parametri nei modelli GARCH è molta vicina all'unità. Invece, nel modello IGARCH consideriamo la somma dei parametri uguale ad uno, il che significa che i rendimenti della serie non stazionari in covarianza e che c'è una radice unitaria nel processo. Quindi, la condizione del modello IGARCH è:

$$\sum_{i=1}^{q} \alpha_i + \sum_{i=1}^{p} \beta_i = 1$$

L'IGARCH (1, 1) è così specificato:

$$h_t = w + \alpha \varepsilon_{t-1}^2 + \beta h_{t-1} \qquad (13)$$

dove $w > 0, \alpha > 0, \beta > 0\ e\ \alpha + \beta = 1$.

1.5 GARCH multivariati

Capire e prevedere la dipendenza temporale nel momento secondo (varianza) dei rendimenti degli assets è molto importante nelle analisi econometriche finanziarie. Analizzare tale momento attraverso modelli multivariati porta a dei risultati empirici maggiormente accurati rispetto a modelli univariati. Questi studi portano ad un miglioramento delle decisioni prese in varie aree, come ad esempio in *asset pricing, portfolio optimization, option pricing, hedging and risk management*. L'utilizzo ovvio degli MGARCH (GARCH multivariati) è lo studio delle relazioni tra le volatilità, covarianze e correlazioni di diversi mercati. I modelli

MGARCH sono stati sviluppati inizialmente verso la fine degli anni '80 e nella prima metà degli anni '90. Tali modelli, poi, sono stati parzialmente coperti in *Franses e van Dijk (2000)* e in *Gourieroux (1997).*

1.5.1 Panoramica del modello

Consideriamo un processo stocastico $\{y_t\}$ di dimensione N x 1. Tale processo stocastico è condizionato da informazioni passate, denotate da I_{t-1}. Definiamo θ un vettore finito di parametri e scriviamo[4]:

$$y_t = \mu_t(\theta) + \varepsilon_t \tag{14}$$

dove $\mu_t(\theta)$ è la media condizionata del vettore e

$$\varepsilon_t = H_t^{1/2}(\theta)\, z_t \tag{15}$$

dove $H_t^{1/2}$ (θ) è una matrice definita positiva N x N. Inoltre, assumiamo che z_t sia un vettore N x 1 ed abbia i seguenti due primi momenti:

$$E(z_t) = 0$$

$$Var(z_t) = I_N \tag{16}$$

dove I_N è una matrice identità di ordine N. Per rendere più chiaro cos'è $H_t^{1/2}$, calcoliamo la matrice di varianza condizionata di y_t:

$$\begin{aligned} Var(y_t|I_{t-1}) &= Var_{t-1}(y_t) = Var_{t-1}(\varepsilon_t) \\ &= H_t^{1/2} Var_{t-1}(z_t)\left(H_t^{1/2}\right) = H_t \end{aligned} \tag{17}$$

Quindi $H_t^{1/2}$ è una matrice definita positiva di dimensioni N x N tale che H_t sia la matrice di varianza condizionata di y_t. Come possiamo notare dalle equazioni (1) e (2), H_t e μ_t dipendono dal vettore di parametri θ.

1.5.2 Modello VEC

La formulazione di un primo modello multivariato fu proposta da *Bollerslev et al. (1988).* Il modello VEC (1,1) è definito come:

$$h_t = c + A\eta_{t-1} + Gh_{t-1} \tag{18}$$

[4] L. Bauwens et al., Multivariate GARCH models: A survey, Journal of Applied Econometrics,21: 79-106 (2006)

dove

$$H_t = vech(H_t) \quad (19)$$

$$\eta_t = vech\,(\varepsilon_t \varepsilon_t') \quad (20)$$

e vech $(*)$ è un operatore che considera solo la parte triangolare inferiore della matrice N x N inserendola all'interno di un vettore di dimensioni N (N+1) / 2 x 1. *A* e *G* sono matrici di parametri quadrate di ordine (N+1) N /2 e *c* è un vettore di parametri di ordine (N+1) N /2 x 1.

I punti deboli del modello sono l'aumento repentino dei parametri da stimare al crescere dei titoli considerati e anche l'impossibilità di vincolare le matrici *A* e *G* in modo tale da far sì che la matrice delle varianze e delle covarianze sia sicuramente definita positiva. Infatti, nel caso di N = 3 il numero di parametri da stimare è 78, se N = 4 dobbiamo stimare 210 parametri e così via. Questo implica che in pratica il modello è utilizzato solo in un caso bivariato. Per superare tali limiti la letteratura ha suggerito un nuovo modello semplificato: il modello GARCH diagonale (DVEC). Il nuovo modello prevede che le matrici *A* e *G* siano diagonali, ogni elemento h_{ijt} dipende solo dal proprio *lag* e dai valori passati del prodotto $\varepsilon_{it}\,\varepsilon_{jt}'$. Questa restrizione riduce il numero di parametri da stimare a N (N+5) /2, ad esempio nel caso di N = 3 i parametri sono 12. Inoltre, vengono poste delle condizioni necessarie e sufficienti che i parametri devono rispettare al fine di garantire che la matrice di varianza condizionata sia definita positiva. Queste condizioni derivano dal prodotto di Hadamard (indicato con il simbolo $\odot$)[5]. Il modello può essere scritto come[6]:

$$H_t = C^{\circ} + A^{\circ} \odot (\varepsilon_{t-1}\varepsilon_{t-1}') + G^{\circ} \odot H_{t-1} \quad (21)$$

dove A = diag [vech (A°)], G = diag [vech (G°)] e c = vech (C°). Per le proprietà del prodotto di Hadamard è semplice da dimostrare che la matrice H_t sia definita positiva per ogni t supposto che le matrici C°, A°, G° siano definite positive.

1.5.3 Modello BEKK

Proposto in Engle e Kroner (1995), il BEKK (1, 1, K) è definito come:

$$H_t = C'C + \sum_{k=1}^{k} A_k' \varepsilon_{t-1}\varepsilon_{t-1}' A_k + \sum_{k=1}^{k} G_k' H_{t-1} G_k \quad (22)$$

dove C, A_k e G_k sono matrici N x N ma C è triangolare superiore. Invece, K determina la generalità del processo[7]. Il numero di parametri in BEKK (1, 1, 1) è N (5N + 1) /2. Per ridurre tale numero può essere imposto un modello diagonale BEKK, dove le matrici A_k e G_k sono matrici diagonali. Un altro modo per ridurre il numero di parametri è quello di utilizzare un modello BEKK scalare, in cui le matrici A_k e G_k sono uguali ad

[5] Per due matrici A e B delle stesse dimensioni m x n, il prodotto di Hadamard (A ⊙ B) è una matrice delle stesse dimensioni degli operandi con gli elementi dati da (a_{ij} b_{ij}).
[6] L. Bauwens et al., Multivariate GARCH models: A survey, Journal of Applied Econometrics,21: 79-106 (2006)
[7] Per evitare che il numero dei parametri sia troppo elevato viene utilizzato K = 1.

uno scalare moltiplicato per una matrice di uno. Nel modello VEC per far sì che sia stazionario in covarianza è richiesto che gli autovalori di A + G siano minori di uno in modulo. La matrice di varianza non condizionata $\sum$ è data da vech ($\sum$) = $[I_{N^*} - A - G]_c^{-1}$, dove $N^* = N(N+1)/2$. Una simile espressione può essere ricavata anche nel caso del modello BEKK. Infatti, è possibile ricondursi ad un modello VEC da uno BEKK applicando l'operatore vec[8] al posto del vech[9]:

$$vec\ (H_t) = W + \sum_{k=1}^{k} \sum_{i=1}^{q} (A'_{ki} \otimes A'_{ki})\ vec\ (\varepsilon_{t-1}\varepsilon'_{t-1}) +$$

$$+ \sum_{k=1}^{K} \sum_{j=1}^{p} (G'_{kJ} \otimes G'_{kj})\ vec\ (H_{t-1}) \qquad (23)$$

dove W = $(C' \otimes C')$ vec (I). Quindi, possiamo affermare che il modello BEKK è stazionario in covarianza quando sono strettamente minori di uno gli autovalori della matrice

$$\sum_{k=1}^{k} \sum_{i=1}^{q} (A'_{ki} \otimes A'_{ki}) + \sum_{k=1}^{K} \sum_{j=1}^{p} (G'_{kJ} \otimes G'_{kj})$$

Per semplificare, chiamiamo $A^* = \sum_{k=1}^{k} \sum_{i=1}^{q} (A'_{ki} \otimes A'_{ki})$ e $G^* = \sum_{k=1}^{K} \sum_{j=1}^{p} (G'_{kJ} \otimes G'_{kj})$, la matrice di varianza non condizionata del modello BEKK è data da:

$$H_t = [I - (A^* + G^*)]^{-1}\ vec(C'C) \qquad (24)$$

1.5.4 Modello GARCH fattoriale

Engle et al. (1990) introdussero un nuovo modello di parametrizzazione di H_t, partendo dall'idea che i "co-movimenti" dei rendimenti dei titoli sono guidati da un piccolo numero di variabili sottostanti comuni, che appunto sono chiamati fattori. Questo modello può essere visto come un particolare BEKK (Lin, 1992).
Il BEKK (1, 1, K) è un modello GARCH fattoriale, indicato come F-GARCH (1, 1, K), se per ogni k = 1,…,K, A_k e G_k hanno rango uno e hanno lo stesso autovettore destro e sinistro, λ_k e w_k, tale che[10]:

$$A_k = \alpha_k w_k \lambda'_k \qquad G_k = \beta_k w_k \lambda'_k \qquad (25)$$

dove α_k e β_k sono scalari, e λ_k e w_k sono vettori N x 1 che per ogni k = 1,…,K soddisfa le seguenti condizioni:

[8] L'operatore vec di una matrice è una trasformazione lineare che converte la matrice in un vettore colonna.
[9] L. Bauwens et al., Multivariate GARCH models: A survey, Journal of Applied Econometrics,21: 79-106 (2006)
[10] L. Bauwens et al., Multivariate GARCH models: A survey, Journal of Applied Econometrics,21: 79-106 (2006)

$$w_k'\lambda_i = \begin{cases} 0 \; per \; k \neq i \\ 1 \; per \; k = i \end{cases} \qquad (26)$$

$$\sum_{n=1}^{N} w_{kn} = 1 \qquad (27)$$

Se sostituiamo la (12) e la (13) nella (9) e definiamo $\Omega = C'C$, otteniamo:

$$H_t = \Omega + \sum_{k=1}^{K} \lambda_k \lambda_k' (\alpha_k^2 w_k' \varepsilon_{t-1} \varepsilon_{t-1}' w_k + \beta_k^2 w_k' H_{t-1} w_k) \qquad (28)$$

1.5.5 Modello con correlazioni condizionali costanti

Questo modello, a differenza dei modelli esposti nei precedenti paragrafi, può essere visto come combinazione non lineare di GARCH univariati. Questo permette di specificare separatamente, da una parte, la matrice di varianza condizionata e, dall'altra, la matrice di correlazione condizionata. Il modello con correlazioni condizionali costanti (CCC) è una prima soluzione a dei procedimenti di stima molto complessi e con un numero elevato di parametri. *Bollerslev (1990)* propone tale modello in cui le correlazioni condizionate sono costanti e quindi le covarianze condizionate sono parametrizzate in modo tale da essere proporzionali al prodotto delle corrispondenti deviazioni standard. In questo modo, oltre a ridurre drasticamente il numero di parametri non conosciuti, semplificando i processi di stima, è possibile ottenere una matrice H_t definita positiva sotto deboli condizioni. Il modello CCC è definito come:

$$H_t = D_t \, R \, D_t = \left(p_{ij} \sqrt{h_{iit} h_{jjt}}\right) \qquad (29)$$

dove

$$D_t = diag\left(h_{11t}^{1/2} \dots h_{NNt}^{1/2}\right) \qquad (30)$$

h_{iit} può essere definito come un qualsiasi GARCH univariato, e R = (ρ_{ij}) è una matrice simmetrica definita positiva con ρ_{ii}=1 ($\forall \; i$). La matrice R, quindi, contiene tutte le correlazioni condizionali costanti. Ogni varianza condizionale contenuta in D_t è specificata da un modello GARCH (1, 1):

$$h_{iit} = w_i + \alpha_i \varepsilon_{i,t-1}^2 + \beta_i h_{ii,t-1} \qquad i = 1, \dots, N \qquad (31)$$

Il modello CCC ha N (N+5) /2 parametri, la matrice H_t è definita positiva se e solo se tutte le varianze condizionali sono positive e se la matrice R è definita positiva. Tuttavia, tale modello è stato oggetto di molte critiche in quanto l'assunzione che le correlazioni condizionali siano costanti nel tempo è irrealistica. Christodoulakis e Satchell (2002), Engle (2002) e Tse e Tsui (2002) proposero una soluzione a tale problema

considerando la matrice delle correlazioni condizionali dipendente dal tempo. Questo modello è definito modello con correlazioni condizionali dinamiche (DCC) che sarà oggetto del prossimo paragrafo.

1.5.6 Modello con correlazioni condizionali dinamiche

Nel modello DCC proposto in Christodoulakis e Satchell (2002) viene utilizzata la trasformazione di Fisher per i coefficienti di correlazione[11]. Questo modello è molto semplice da implementare perché la positività della matrice delle correlazioni condizionali è garantita dalla trasformazione di Fisher. Tuttavia, tale modello è utilizzato solo nel caso bivariato.
Il modello DCC proposto in Tse e Tsui (2002), invece, è definito da[12]:

$$H_t = D_t R_t D_t \tag{32}$$

dove D_t è definite come nella (17), h_{iit} può essere specificata da un qualsiasi modello GARCH univariato e

$$R_t = (1 - \theta_1 - \theta_2)R + \theta_1 \psi_{t-1} + \theta_2 R_{t-1} \tag{33}$$

Nella (20), θ_1 e θ_2 sono parametri non negativi tali per cui $\theta_1 + \theta_2 < 1$, R è una matrice simmetrica definita positiva N x N con $\rho_{ii} = 1$ e ψ_{t-1} è la matrice delle correlazioni N x N di ε_t per $\tau = t - M, t - M + 1, \ldots, t - 1$. I suoi elementi sono dati da:

$$\psi_{ij,t-1} = \frac{\sum_{m=1}^{M} u_{i,t-m} u_{j,t-m}}{\sqrt{\left(\sum_{m=1}^{M} u_{i,t-m}^2\right)\left(\sum_{m=1}^{M} u_{j,t-m}^2\right)}} \tag{34}$$

dove $u_{it}/\sqrt{h_{iit}}$. La matrice ψ_{t-1} può essere espressa come:

$$\psi_{t-1} = B_{t-1}^{-1} L_{t-1} L_{t-1}' B_{t-1}^{-1} \tag{35}$$

dove B_{t-1} è una matrice diagonale N x N con l'i-esimo elemento della diagonale dato da $\left(\sum_{h=1}^{M} u_{i,t-h}^2\right)^{1/2}$ e $L_{t-1} = (u_{t-1}, \ldots, u_{t-M})$ è una matrice N x M con $u_t = (u_{1t} u_{2t} \ldots u_{Nt})'$. Una condizione necessaria per assicurare la positività di ψ_{t-1}, di cui alla (22), e quindi anche di R_t (vd. 20), è che $M \geq N$.

[11] La specificazione del coefficiente di correlazione è $\rho_{12,t} = (e^{2r_t} - 1)/(e^{2r_t} + 1)$ dove r_t può essere definito con un qualsiasi modello GARCH utilizzando $\varepsilon_{1t}\varepsilon_{2t}/\sqrt{h_{11t}h_{22t}}$ come errore (L. Bauwens et al., Multivariate GARCH models: A survey, Journal of Applied Econometrics,21: 79-106 (2006)).
[12] L. Bauwens et al., Multivariate GARCH models: A survey, Journal of Applied Econometrics,21: 79-106 (2006)

Infine, illustriamo il modello DCC proposto da Engle (2002) che è definito come nella (19) con[13]:

$$R_t = \tilde{Q}_t^{-1} Q_t \tilde{Q}_t^{-1} \quad (36)$$

dove $\tilde{Q}_t$[14] è una matrice diagonale con le radici quadrate degli elementi posti sulla diagonale di Q_t. La matrice simmetrica Q_t (N x N) è data da:

$$Q_t = (1 - \alpha - \beta)\bar{Q} + \alpha u_{t-1} u'_{t-1} + \beta Q_{t-1} \quad (37)$$

con $u_t = u_{it}/\sqrt{h_{iit}}$. $\bar{Q}$ è la matrice (N x N) di varianza non condizionale di u_t, α e β sono parametri non negativi tali per cui $\alpha + \beta < 1$.

Uno sconveniente dei modelli DCC è che θ_1, θ_2, nel caso del modello proposto in Tse e Tsui (2002), e α, β (Engle (2002)) sono degli scalari, cosicché tutte le correlazioni condizionali seguono lo stesso processo. Questo è necessario per assicurare che R_t sia definita positiva ∀t attraverso condizioni sufficienti sui parametri. I modelli DCC possono essere stimati in due fasi, questo approccio rende il processo di stima meno complesso nel caso in cui N sia alto.

1.5.7 Modello O-GARCH

Il modello GARCH ortogonale viene proposto in Kariya (1988) e in Alexander e Chibumba (1997), considerato come una combinazione lineare di diversi modelli GARCH univariati. In particolare, questo modello ha tale denominazione perché si assume che le osservazioni siano generate da una trasformazione ortogonale di N modelli GARCH univariati. Il modello O-GARCH è definito come[15]:

$$V^{-1/2}\varepsilon_t = \mu_t = \Lambda_m f_t \quad (38)$$

dove $V = diag(v_1, v_2, \dots, v_N)$, con v_i la varianza di ε_{it}, e Λ_m è una matrice di dimensioni N x m data da:

$$\Lambda_m = P_m diag\left(l_1^{1/2} \dots l_m^{1/2}\right) \quad (39)$$

dove l_i $(l_1 \geq \dots \geq l_m > 0)$ sono gli autovalori della matrice μ_t e P_m è la matrice N x m degli autovettori associati. Il vettore $f_t = (f_{1t} \dots f_{mt})'$ è definito in modo tale che:

---

[13] L. Bauwens et al., Multivariate GARCH models: A survey, Journal of Applied Econometrics,21: 79-106 (2006)

[14] $\tilde{Q}_t = diag\left(q_{11,t}^{1/2} \dots q_{NN,,t}^{1/2}\right)$ dove $Q_t = (q_{ij,t})$; $\tilde{Q}_t$ viene inserita nel modello per assicurare che R_t sia effettivamente una matrice delle correlazioni (Billio, Caporini e Gobbo (2003)).

[15] L. Bauwens et al., Multivariate GARCH models: A survey, Journal of Applied Econometrics,21: 79-106 (2006)

$$E_{t-1}(f_t) = 0 \qquad Var_{t-1}(f_t) = \Sigma_t = diag\left(\sigma_{f1t}^2, \dots, \sigma_{fmt}^2\right) \tag{40}$$

$$\sigma_{fit}^2 = (1 - \alpha_i - \beta_i) + \alpha_i f_{i,t-1}^2 + \beta_i \sigma_{f_{i,t-1}}^2 \quad i = 1, \dots, m \tag{41}$$

Di conseguenza:

$$H_t = Var_{t-1}(\varepsilon_t) = V^{1/2} V_t V^{1/2} \qquad V_t = Var_{t-1}(\mu_t) = \Lambda_m \Sigma_t \Lambda_m' \tag{42}$$

Il modello O-GARCH, quindi, può essere considerato come un'alternativa ai modelli CCC e DCC in quanto in grado di analizzare la volatilità anche in presenza di un elevato numero di asset. Tuttavia, dato che la condizione di ortogonalità della matrice è molto restrittiva, viene implementato in Van der Weide (2002) un nuovo modello (GO-GARCH) dove viene assunto che la matrice Λ nella relazione $\mu_t = \Lambda_m f_t$ sia quadrata e invertibile piuttosto che ortogonale.

1.6 Processo di stima

Nelle precedenti sezioni abbiamo introdotto i vari modelli GARCH univariati e multivariati utilizzati per modellizzare il momento secondo dei rendimenti degli assets. In questa sezione introdurremo in che modo i parametri di tali modelli vengono stimati, e l'approccio a due passaggi utilizzato nei modelli DCC.

1.6.1 Stima ML

Supponiamo di avere un processo stocastico $\{y_t\}$ di cui media condizionale, varianza condizionale e distribuzione condizionale sono rispettivamente $\mu_t(\theta_0), H_t(\theta_0)$ *e* $p(y_t|\zeta_0, I_{t-1})$, dove $\zeta_0 = (\theta_0 \eta_0)$ è un vettore di parametri e η_0 è un vettore contenente i parametri della distribuzione di z_t (L. Bauwens et al. (2006)). La procedura di stima spesso usata è quella che involve la massimizzazione della funzione di verosimiglianza partendo dall'ipotesi che la componente di innovazione standardizzata z_t $(\varepsilon_t(\eta)/\sigma_t(\eta))$ sia una variabile i.i.d. La funzione di densità della componente di innovazione è data da $g(z_t(\theta)|\eta)$ dove η è il vettore dei parametri di disturbo. Utilizzando la notazione in *L. Bauwens et al. (2006)*, il problema da risolvere è quello di trovare il massimo della funzione log-verosimiglianza $L_T(\theta, \eta)$ per T osservazioni rispetto ai parametri $\zeta = (\theta, \eta)$:

$$L_T(\zeta) = \sum_{t=1}^{T} log\, f\,(y_t|\zeta, I_{t-1}) \tag{43}$$

$$f(y_t|\zeta, I_{t-1}) = |H_t|^{-1/2} g\left(H_t^{-1/2}(y_t - \mu_t|\eta)\right) \tag{44}$$

Il termine $|H_t|^{-1/2}$ è il determinante del Jacobiano ottenuto nel passaggio dall'innovazione alla standardizzata. La distribuzione più comune presente nella letteratura è la normale multivariata, determinata unicamente dai suoi primi due momenti. In questo caso la funzione di log-verosimiglianza diventa:

$$L_T(\theta) = -\frac{1}{2}\sum_{t=1}^{T} log|H_t| - \frac{1}{2}\sum_{t=1}^{T}(y_t - \mu_t)' H_t^{-1}(y_t - \mu_t) \qquad (45)$$

Tuttavia, è difficile giustificare l'assunzione di normalità dell'innovazione nel caso in cui si sta lavorando con dati giornalieri e settimanali. In particolare, per la presenza di leptocurtosi[16] in molte serie storiche finanziarie. Nonostante tali problematiche, è stato dimostrato in molti studi che dalla massimizzazione della (32) può essere ricavato uno stimatore consistente di θ_0. Ad esempio, in Bollerslev e Wooldridge (1992) questo stimatore, chiamato quasi-massima verosimiglianza (QML), è consistente purché la media e la varianza condizionale siano specificate correttamente.

Un'alternativa alla Gaussiana è la t di Student che ha un parametro extra, gradi di libertà (indicato con υ). Quando questo parametro tende ad infinito, la t di Student tende ad una normale. Quando tale parametro tende a zero, le code diventano sempre più spesse. Inoltre, i gradi libertà indicano anche l'ordine di esistenza dei momenti, ad esempio se $\upsilon = 2$ allora il secondo momento non esiste. Per tale ragione si suppone che υ sia maggiore strettamente di 2. La densità della t di Student può essere definita come:

$$g(z_t|\theta,\upsilon) = \frac{\Gamma\left(\frac{\upsilon+N}{2}\right)}{\Gamma\left(\frac{\upsilon}{2}\right)[\pi(\upsilon-2)]^{\frac{N}{2}}}\left[1 + \frac{z_t' z_t}{\upsilon-2}\right]^{-\frac{N+\upsilon}{2}} \qquad (46)$$

dove $\Gamma(\cdot)$ è la funzione Gamma. Tuttavia, potrebbe succedere che con la t di Student si debba far fronte sia alla skewness che ad un eccesso di curtosi. In alcuni studi viene mostrato come si possa trattare tale problema combinando un GARCH multivariato ad una densità multivariata per le innovazioni. Le densità utilizzate sono misture di densità di normali multivariate, come ad esempio la densità skew-student multivariata (vd. Bauwens e Laurent, 2002).

1.6.2 Stima modello DCC

Come già detto in precedenza, il modello DCC può essere stimato utilizzando un approccio a due passaggi e questo è molto utile nel caso di N elevato. In Engle e Sheppard (2001) viene mostrato come la log-verosimiglianza possa essere scritta come una somma della parte di media e volatilità, che dipendono da parametri θ_1, e della parte di correlazione dipendente da parametri θ_2. Uno stimatore consistente del parametro θ_1 può essere ricavato sostituendo R_t (vd. (19)) con la matrice identità nella (32). In questo caso, la funzione

[16] Si dice che una distribuzione è leptocurtica quando la curtosi è maggiore di 3 cioè quando presenta troppi valori estremi per essere assimilati ad una normale.

di quasi-log-verosimiglianza è uguale ad una somma di funzioni di log-verosimiglianza di N modelli univariati[17]:

$$QL1_T(\theta_1) = -\frac{1}{2}\sum_{t=1}^{T}\sum_{i=1}^{N}\left[log(h_{iit}) + \frac{(y_{it}-\mu_{it})^2}{h_{iit}}\right] \quad (47)$$

Invece, uno stimatore consistente di θ_2 può essere ricavato dalla massimizzazione di:

$$QL2_T(\theta_2|\theta_1) = -\frac{1}{2}\sum_{t=1}^{T}(log|R_t| + u_t' R_t^{-1} u_t) \quad (48)$$

dove $u_t = D_t^{-1}(y_t - \mu_t)$. La somma delle funzioni di verosimiglianza nella (34) e (35), più il semiprodotto della somma totale dei residui standardizzati al quadrato è uguale alla funzione di log-verosimiglianza nella (32). Quindi, è possibile fare una comparazione tra le funzioni ricavate con un approccio a due passaggi e quelle con un solo passaggio.

2 Modellando la volatilità dei rendimenti di criptovalute usando modelli GARCH

Il prezzo del Bitcoin, e come altre criptovalute, è conosciuto per essere molto volatile. Gli investitori sono molto interessati a stimare accuratamente la volatilità di queste criptovalute. È noto che la volatilità non è osservabile quindi bisogna utilizzare modelli efficienti che possono accuratamente cogliere tale volatilità dal prezzo. Il mercato delle criptovalute è relativamente nuovo quindi non c'è molta letteratura sui modelli utilizzati per stimare la volatilità del mercato. In questo capitolo si andrà a specificare la media e la volatilità dei rendimenti di criptovalute al fine di costruire un portafoglio. Le criptovalute utilizzate in questa analisi sono: Bitcoin, Litecoin, Ethereum e Monero (valuta in USD). Nei primi paragrafi di questo capitolo si illustrerà l'analisi sull'univariato del bitcoin per capire quale sia il modello GARCH migliore. Successivamente, l'analisi passerà al multivariato utilizzando il modello DCC. Tutte le analisi vengono svolte con l'aiuto di R e i codici di programmazione utilizzati sono in *Appendice*.

2.1 Data

In questa analisi il prezzo di chiusura della criptovaluta è considerato come prezzo della criptovaluta perché esso incorpora tutte le attività del giorno. Le serie storiche delle criptovalute vengono esportate direttamente da https://it.finance.yahoo.com/ tramite una funzione utilizzata su R (quantmod package). Le serie storiche esportate partono dal 07/08/2015 al 01/12/2019. Dato che non è possibile lavorare sui prezzi perché, come è facile da intuire dalla Figura 2, la serie storica dei prezzi non è stazionaria, bisognerà utilizzare i rendimenti come nostra variabile. Per convincersi maggiormente, guardiamo alla Figura 3 in cui viene raffigurato il

[17] L. Bauwens et al., Multivariate GARCH models: A survey, Journal of Applied Econometrics,21: 79-106 (2006)

grafico delle autocorrelazioni. Dal grafico dell'ACF vediamo che le autocorrelazioni sono molto significative presentando forti dipendenze lineari e quindi possiamo confermare quanto già accennato, ovvero che la serie è non stazionaria. Assumendo che P_t e P_{t-1} rappresentino rispettivamente il prezzo corrente e del giorno precedente della criptovaluta, i rendimenti della serie sono così calcolati:

$$R_t = log(P_t) - log(P_{t-1}) \qquad (49)$$

Sulla base di un'ispezione visiva dell'oggetto della serie temporale dei rendimenti in Figura 4 è possibile affermare che la serie sia stazionaria, ma nel prossimo paragrafo verranno effettuati alcuni test per confermare quanto accennato. Intanto, soffermiamo l'attenzione su alcune delle statistiche descrittive della serie (Figura 5). La funzione "*basicStats*" ci fornisce una sintesi di tutte le statistiche descrittive della serie. La *skewness* ci dà indicazione di quanto questa distribuzione si concentri attorno alla propria media oppure si disperda a destra o a sinistra di questa. In particolare, nel nostro caso, abbiamo una *skewness* prossima allo 0, con un valore approssimativo di 0,2. Nel caso in esame, una *skewness positiva* (maggiore di 0), anche se di poco, rappresenta una situazione negativa in quanto va a significare che la distribuzione delle *performances* giornaliere si concentra a sinistra della media. La *kurtosis*, invece, è un indice che misura lo spessore delle code di una funzione di densità. Se il coefficiente di *kurtosis* è minore di 0, la curva si definisce *platicurtica*, cioè più piatta di una normale, e sta a significare che la dispersione dei valori intorno alla propria media è molto ampia. Se il coefficiente di kurtosis è maggiore di 0, come nel nostro caso, la curva si definisce leptocurtica e i valori della distribuzione sono concentrati intorno alla media. La leptocurtosi è fenomeno comune a tutte o quasi tutte le serie storiche dei rendimenti di asset finanziari.

Figura 2: Serie storica prezzo Bitcoin 07/08/2015 – 01/12/2019

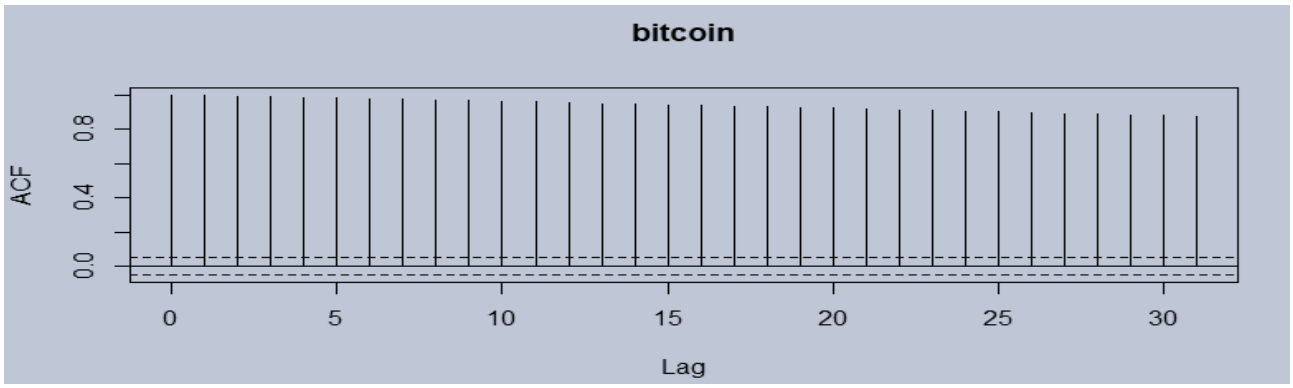

Figura 3: Grafico ACF Bitcoin

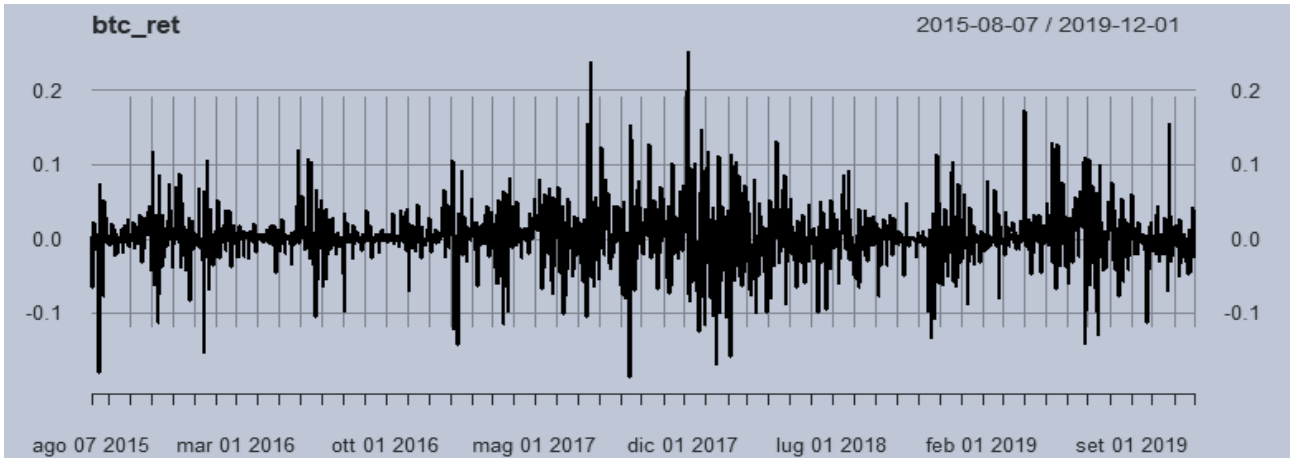

Figura 5: Serie storica rendimenti 07/08/2015 – 01/12/2019

	btc_ret
nobs	1578.000000
NAs	0.000000
Minimum	-0.187411
Maximum	0.252472
1. Quartile	-0.011279
3. Quartile	0.017878
Mean	0.002853
Median	0.002202
Sum	4.502683
SE Mean	0.000990
LCL Mean	0.000911
UCL Mean	0.004796
Variance	0.001548
Stdev	0.039339
Skewness	0.204641
Kurtosis	4.923186

Figura 4: Statistiche descrittive

2.2 Test di stazionarietà

Come già accennato, una serie storica y_t è detta stazionaria se le sue proprietà statistiche (media, varianza e autocorrelazione) non variano nel tempo. Inoltre, si è detto che da una ispezione visiva dell'oggetto grafico

della serie temporale è possibile capire se sia stazionaria o meno. Tuttavia, è possibile usufruire dell'aiuto di alcuni test di stazionarietà che ci permettono di capire se una serie sia stazionaria o meno. I test utilizzati in questo studio sono: *augmented Dickey-Fuller* (ADF) e *Phillips-Perron* (PP). Entrambi hanno come ipotesi nulla e alternativa le seguenti:

- H_0: la serie non è stazionaria quindi presenta una radice unitaria
- H_1: la serie è stazionaria quindi non ci sono radici unitarie al suo interno

Come è possibile verificare dalla Figura 6, entrambi i test confermano quanto già detto sulla base di una ispezione visiva della serie, ovvero che la serie è stazionaria.

```
        Augmented Dickey-Fuller Test

data:  btc_ret
Dickey-Fuller = -10.621, Lag order = 11, p-value = 0.01
alternative hypothesis: stationary
```

```
      Phillips-Perron Unit Root Test

data:  btc_ret
Dickey-Fuller = -39.627, Truncation lag parameter = 7, p-value = 0.01
```

Figura 6: Test di stazionarietà

2.3 Arch effects

Per applicare dei modelli GARCH alla serie dei rendimenti dobbiamo testare la presenza di "*ARCH effects"* sui residui della serie. Per fare ciò, definiamo un modello ARIMA (p, d, q) per la nostra serie. Sulla base delle autocorrelazioni (e autocorrelazioni parziali, Figure 7 e 8) e del criterio di Akaike[18] (Tab. 1), scegliamo un modello ARIMA (3, 0, 3).

Inoltre, i parametri di questo modello sono tutti molto significativi tranne la costante (Figura 9).

Specificato il modello, testiamo la presenza di *"ARCH effects"* tramite l'utilizzo del *Ljung Box test* che va a verificare se ci sono autocorrelazioni seriali nei residui al quadrato del nostro modello. Quindi, il *Ljung Box test* è un metodo usato per testare l'assenza di autocorrelazioni seriali fino ad un determinato lag *k*. Può essere definito come:

- H_0: I dati osservati non presentano autocorrelazioni seriali
- H_1: I dati osservati presentano autocorrelazioni seriali

Il test statistico è dato da:

$$Q = N(N+2)\sum_{k=1}^{h}\frac{\hat{\rho}_k^2}{N-k} \qquad (50)$$

dove N è la numerosità del campione, h il numero dei *lag*, $\hat{\rho}_k^2$ è l'autocorrelazione della serie stimata al lag *k*. Il Ljung box test sui residui al quadrato è molto significativo (figura 10), quindi possiamo affermare che vi sia

[18] *Akaike information criterion* (AIC) stima la qualità di ciascun modello relativamente ad ogni altro modello (viene scelto il modello che presenta l'akaike minore).

significativa autocorrelazione e di conseguenza la presenza di ARCH effects. Questo è possibile osservarlo anche dalla figura 11.

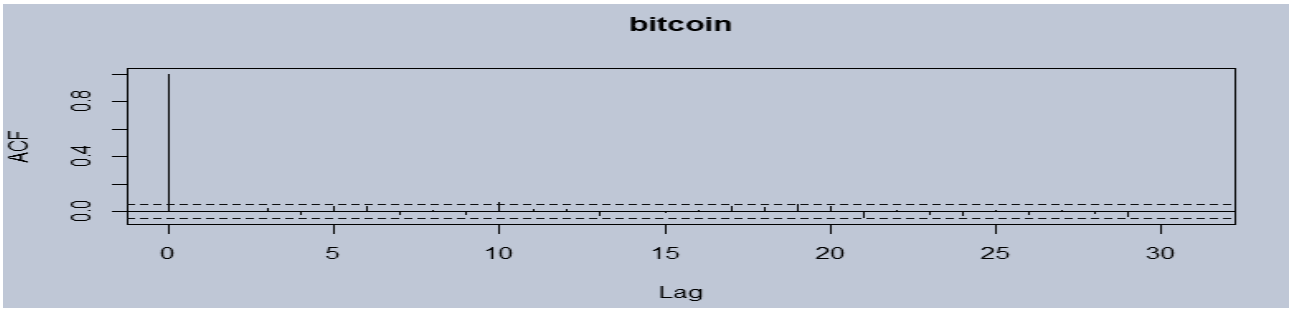

Figura 7: ACF rendimenti Bitcoin

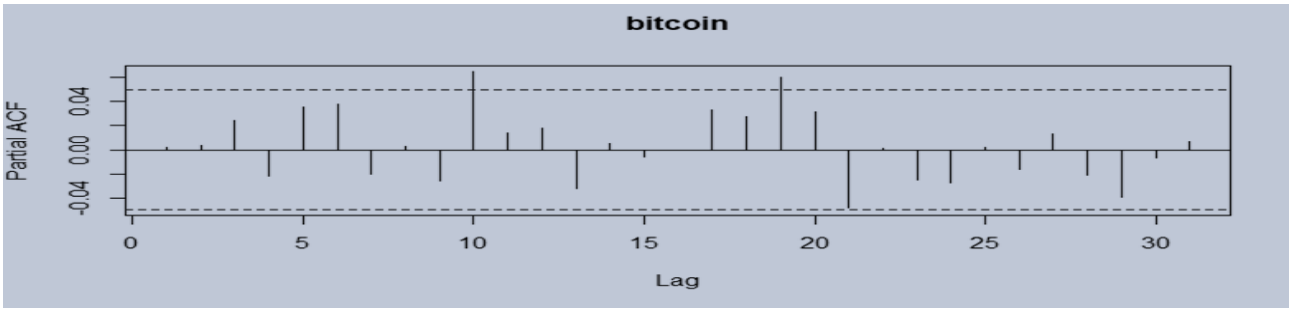

Figura 8: PACF rendimenti Bitcoin

ARIMA (p,d,q)	AIC
(1,0,0)	-5728,21
(2,0,0)	-5726,23
(3,0,0)	-5725,19
(1,0,1)	-5726,21
(2,0,1)	-5724,21
(3,0,1)	-5723,94
(1,0,2)	-5724,23
(2,0,2)	-5722,23
(3,0,2)	-5721,67
(1,0,3)	-5723,74
(2,0,3)	-5721,52
(3,0,3)	-5731,07
(0,0,1)	-5728,21
(0,0,2)	-5726,23
(0,0,3)	-5725,12

Tabella 1: Akaike information criterion dei modelli

```
           Estimate Std. Error  z value Pr(>|z|)
ar1       -0.7228768  0.0062255 -116.115   <2e-16 ***
ar2        0.7114251  0.0114417   62.178   <2e-16 ***
ar3        0.9886593  0.0089837  110.051   <2e-16 ***
ma1        0.7367343  0.0354295   20.794   <2e-16 ***
ma2       -0.7068827  0.0292448  -24.171   <2e-16 ***
ma3       -0.9885024  0.0126435  -78.183   <2e-16 ***
intercept  0.0026663  0.0017235    1.547   0.1219
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

Figura 9: Stima modello ARIMA (3, 0, 3)

```
        Box-Ljung test

data:  residuals_btc_square
X-squared = 270.81, df = 20, p-value < 2.2e-16
```

Figura 10: Box-Ljung test sui residui al quadrato

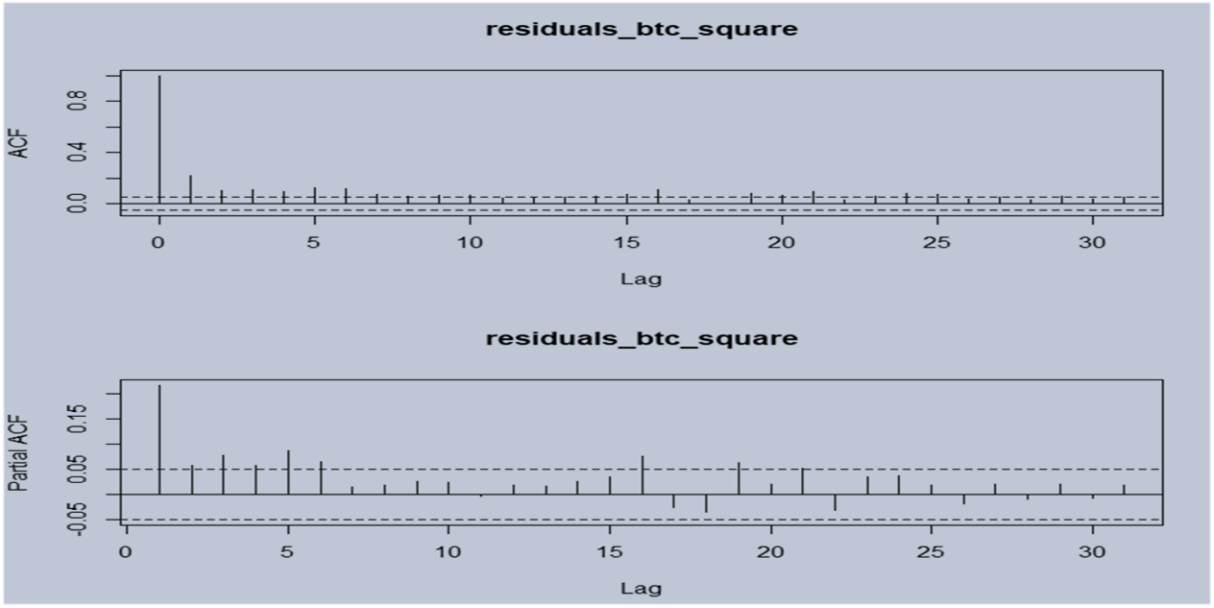

Figura 11: ACF e PACF dei residui al quadrato

2.4 Specificazione modello per la volatilità

In questo paragrafo andremo ad applicare sui residui del modello ARIMA (3, 0, 3) differenti modelli GARCH usando a sua volta differenti distribuzioni. Per tenere conto dell'eccesso di curtosi e delle code grasse che sono presenti nei residui dei rendimenti della serie (vd. Figura 12), utilizziamo tre diversi tipi di distribuzioni che sono: t di student (STD), *generalized error distribution (GED)*, normale inversa gaussiana (NIG).

- La t di student è una sottoclasse di distribuzioni di probabilità continue. Viene spesso utilizzata quando il campionario è piccolo e la standard deviation è sconosciuta. La funzione di densità di probabilità è così definita:

$$f_t(x;\eta) = \frac{\Gamma\left(\frac{\eta+1}{2}\right)}{\sqrt{\eta\pi}\Gamma\left(\frac{\eta}{2}\right)}\left(1+\frac{x^2}{\eta}\right)^{-\frac{\eta+1}{2}},\ x \in (-\infty,\infty)$$

dove η sono i gradi di libertà e $\Gamma(\cdot)$ è la funzione Gamma.

- La *Generalized Error distribution (GED)* fa parte della famiglia delle distribuzioni simmetriche con parametro ν. La funzione di densità di probabilità è così definita:

$$f_{GED}(x;\nu) = k(\nu)exp\left\{-\frac{1}{2}\left|\frac{x}{\lambda(\nu)}\right|^{\nu}\right\}, x \in (-\infty,\infty)$$

dove ν determina il peso delle code, $k(\nu)\ e\ \lambda(\nu)$ sono invece delle costanti e sono così definite:

$$k(\nu) = \frac{\nu}{\lambda(\nu)2^{1+\frac{1}{\nu}}\Gamma\left(\frac{1}{\nu}\right)}$$

$$\lambda(\nu) = \left\{\frac{2^{-\frac{2}{\nu}}\Gamma\left(\frac{1}{\nu}\right)}{\Gamma\left(\frac{3}{\nu}\right)}\right\}^{1/2}$$

- La normale inversa gaussiana è una sottoclasse di distribuzioni iperboliche generalizzate. La funzione di densità di probabilità è definita come:

$$f_{NIG}(x:\alpha,\kappa,\mu,\delta) = \frac{\alpha\delta}{\pi}exp\left\{\delta\sqrt{\alpha^2-\beta^2}+\beta(x-\mu)\right\}K_1[\alpha q(x)]$$

dove $q(x) = ((x-\mu)^2+\delta^2)^{1/2}$. La normale inversa gaussiana ha delle code più pesanti rispetto ad una normale, ed è appropriata per data set con skewness e con code grosse o pesanti.

	residuals_btc_square
nobs	1578.000000
NAs	0.000000
Minimum	0.000000
Maximum	0.056314
1. Quartile	0.000031
3. Quartile	0.001181
Mean	0.001530
Median	0.000203
Sum	2.414493
SE Mean	0.000099
LCL Mean	0.001335
UCL Mean	0.001725
Variance	0.000016
Stdev	0.003947
Skewness	6.393406
Kurtosis	60.300871

Figura 12: Curtosi e Skewness residui dei residui al quadrato

Utilizziamo diversi GARCH nella stima della volatilità quali SGARCH (lo standard GARCH della forma $h_t = w + \alpha\varepsilon_{t-1}^2 + \beta h_{t-1}$), EGARCH, IGARCH, GJR-GARCH, TGARCH (con p=1, q=1). Come possiamo notare dalla *Tabella 2*, i modelli GARCH che utilizzano una distribuzione normale inversa gaussiana sono i più efficienti sulla base dei criteri di valutazione utilizzati (Akaike (AIC), Bayes (BIC), Shibata, Hannan-Quinn) tranne per quanto riguarda il modello IGARCH il quale fitta meglio i dati nel caso di una *generalized error distribution* (GED). Tuttavia, il modello migliore è un EGARCH con distribuzione NIG anche se di poco rispetto ad uno GJR-GARCH. Infatti, entrambi sono modelli con asimmetria che riflettono appunto la natura asimmetrica della risposta degli investitori ai rendimenti azionari e di indici che portano a shock positivi o negativi che hanno un impatto diverso sulla varianza condizionata. Nonostante ciò, i residui dei modelli EGARCH e GJR-GARCH presentano delle correlazioni seriali. Infatti, come è possibile vedere dalle figure 13 e 14, i p-values del Ljung-Box test sono molto piccoli per cui bisogna rifiutare l'ipotesi nulla di assenza di correlazioni seriali. Gli unici modelli (tra quelli selezionati come migliori in base ai criteri AIC, BIC, Shibata, Hannan-Quinn) che non presentano correlazioni seriali nei residui sono l'IGARCH (GED) e il TGARCH (NIG). Infine, scegliamo il modello TGARCH (NIG) il cui modello stimato è in figura 15. Il coefficiente GARCH (β), a differenza di α, è molto significativo ed ha un valore di ca. 0,85 il che sta a significare una buona persistenza nella volatilità. Il parametro *"eta"* è statisticamente non significativo, il
che indica assenza di effetti leverage. I coefficienti *skew e shape* sono i parametri della distribuzione utilizzata e sono statisticamente significativi.

```
Weighted Ljung-Box Test on Standardized Residuals
------------------------------------
                        statistic p-value
Lag[1]                      71.75       0
Lag[2*(p+q)+(p+q)-1][2]     81.16       0
Lag[4*(p+q)+(p+q)-1][5]    109.35       0
d.o.f=0
H0 : No serial correlation

Weighted Ljung-Box Test on Standardized Squared Residuals
------------------------------------
                        statistic   p-value
Lag[1]                      52.35 4.637e-13
Lag[2*(p+q)+(p+q)-1][5]     55.67 1.443e-15
Lag[4*(p+q)+(p+q)-1][9]     57.05 9.548e-15
d.o.f=2
```

Figura 13: Test sui residui, modello EGARCH (distribuzione NIG)

```
Weighted Ljung-Box Test on Standardized Residuals
------------------------------------
                        statistic  p-value
Lag[1]                      68.46 1.11e-16
Lag[2*(p+q)+(p+q)-1][2]     76.25 0.00e+00
Lag[4*(p+q)+(p+q)-1][5]    101.34 0.00e+00
d.o.f=0
H0 : No serial correlation

Weighted Ljung-Box Test on Standardized Squared Residuals
------------------------------------
                        statistic   p-value
Lag[1]                      57.70 3.053e-14
Lag[2*(p+q)+(p+q)-1][5]     61.56 0.000e+00
Lag[4*(p+q)+(p+q)-1][9]     63.24 1.110e-16
d.o.f=2
```

Figura 14: Test sui residui, modello GJR-GARCH (distribuzione NIG)

Modello GARCH	AIC	BIC	Shibata	Hannan-Quinn
SGARCH (t-stud)	-10,626	-10,609	-10,626	-10,620
SGARCH (GED)	-11,459	-11,442	-11,459	-11,453
SGARCH (NIG)	-11,497	-11,477	-11,497	-11,489
EGARCH (t-stud)	-10,668	-10,648	-10,668	-10,660
EGARCH (GED)	-11,463	-11,443	-11,463	-11,456
EGARCH (NIG)	-11,864	-11,840	-11,864	-11,855
IGARCH (t-stud)	-10,732	-10,718	-10,732	-10,727
IGARCH (GED)	-11,444	-11,430	-11,444	-11,439
IGARCH (NIG)	-11,191	-11,174	-11,191	-11,185
GJR-GARCH (t-stud)	-10,735	-10,715	-10,735	-10,727
GJR-GARCH (GED)	N/A	N/A	N/A	N/A
GJR-GARCH (NIG)	-11,746	-11,722	-11,746	-11,737
TGARCH (t-stud)	-10,798	-10,778	-10,798	-10,791
TGARCH (GED)	-1,6103	-1,5899	-1,6103	-1,6027
TGARCH (NIG)	-11,278	-11,255	-11,278	-11,269

Tabella 2: Criteri di valutazione sui modelli stimati

```
GARCH Model      : fGARCH(1,1)
fGARCH Sub-Model      : TGARCH
Mean Model       : ARFIMA(0,0,0)
Distribution     : nig

Optimal Parameters
------------------------------------
        Estimate  Std. Error    t value Pr(>|t|)
mu      0.000007    0.000003    2.19573 0.028111
omega   0.000023    0.000007    3.02194 0.002512
alpha1  0.212481    0.459783    0.46213 0.643985
beta1   0.848521    0.016418   51.68212 0.000000
eta11  -0.891244    4.101342   -0.21731 0.827970
skew   -0.427552    0.015440  -27.69069 0.000000
shape   0.010000    0.002280    4.38674 0.000012

Robust Standard Errors:
        Estimate  Std. Error  t value Pr(>|t|)
mu      0.000007    0.000005  1.27329 0.202916
omega   0.000023    0.000015  1.49302 0.135431
alpha1  0.212481    0.215571  0.98567 0.324296
beta1   0.848521    0.032177 26.37076 0.000000
eta11  -0.891244    1.450205 -0.61457 0.538842
skew   -0.427552    0.075457 -5.66620 0.000000
shape   0.010000    0.005393  1.85412 0.063722

LogLikelihood : 8905.595

Information Criteria
------------------------------------

Akaike       -11.278
Bayes        -11.255
Shibata      -11.278
Hannan-Quinn -11.269
```

Figura 15: Stima modello TGARCH (NIG)

2.5 Multivariato

Passando al caso multivariato, analizzeremo tre serie storiche: Bitcoin, Litecoin, Ethereum e Monero. Come nell'univariato, calcoleremo i rendimenti delle serie dei prezzi secondo la (48). In figura 16, 17 e 18 troviamo le serie dei prezzi e dei rendimenti dell'Ethereum, Litecoin e di Monero. Mentre, in figura 19 abbiamo le statistiche descrittive delle tre serie storiche. Possiamo notare come, anche in questo caso, le tre serie finanziare presentano un eccesso di curtosi e di skewness. Prima di procedere con l'analisi multivariata, verifichiamo per la stazionarietà delle tre serie temporali (cfr. par. 2.2). Entrambi i test ADF e PP ci dicono le serie sono stazionarie ed è quindi possibile procedere con l'analisi. Dato che le serie storiche da analizzare sono quattro, avremo una matrice dei rendimenti sulla quale specificheremo un modello VAR (*vector autoregressive*) per l'equazione della media. Il *vector autoregressive model* (VAR) estende l'idea di autoregressione univariata alle regressioni di *k* serie temporali, dove i valori passati (*lagged values*) di tutte le *k* serie appaiono come regressori. In altre parole, in un modello VAR regrediamo un vettore di variabili delle serie temporali sui vettori di queste variabili "ritardati" di un periodo (un periodo nel caso in cui p=1, VAR (p)). Sia $\{y_t = (y_{1t}, \dots, y_{Kt})'; t \in Z\}$ un processo stocastico, diciamo che y_t segue un *vector autoregressive model* di ordine p, connotato come VAR (p) se:

$$y_t = c + A_1 y_{t-1} + \cdots + A_p y_{t-p} + \varepsilon_t, \qquad t \in Z$$

- p è un intero positivo, indica l'ordine del processo ovvero di quanti periodi sono "ritardate" le variabili;
- A_i sono le matrici dei coefficienti (K x K);
- $c = (c_1, \dots, c_K)'$ è un vettore (K x 1) contenente le costanti del modello;
- $\varepsilon_t = (\varepsilon_{1t}, \dots, \varepsilon_{Kt})'$ sono i residui del modello con matrice di covarianza Σ_ε (si presume che la matrice di covarianza non sia singolare, ovvero sia una matrice invertibile).

Il modello VAR (1) stimato presenta un AIC = -23.83691 e un BIC = -23.76893 però ha solo pochi parametri significativi, ciò sta a significare che le variabili non hanno effetti di causalità tra di loro. Questo può essere confermato anche utilizzando il test di Granger-Causality. Il test di Granger-Causality va appunto a testare l'ipotesi nulla H_0 per cui la serie storica (x) non influenza la serie storica (y). I risultati dei test presentano dei p-value grandi per cui non è possibile rifiutare l'ipotesi nulla e quindi, in termini semplici, i valori passati di una delle variabili non è utile a predire i valori delle altre. Tutto ciò non cambia anche utilizzando dei VAR con p > 1 (VAR (2), VAR (3) e così via). Il modello stimato è mostrato in figura 20.

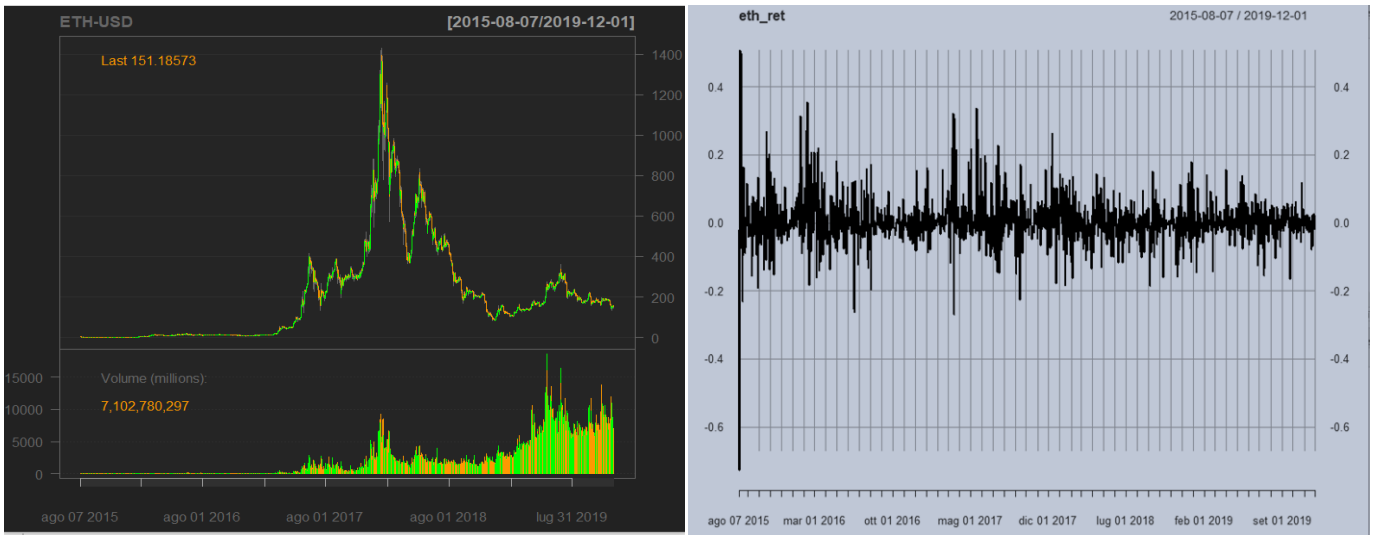

Figura 16: Serie prezzi e rendimenti Ethereum

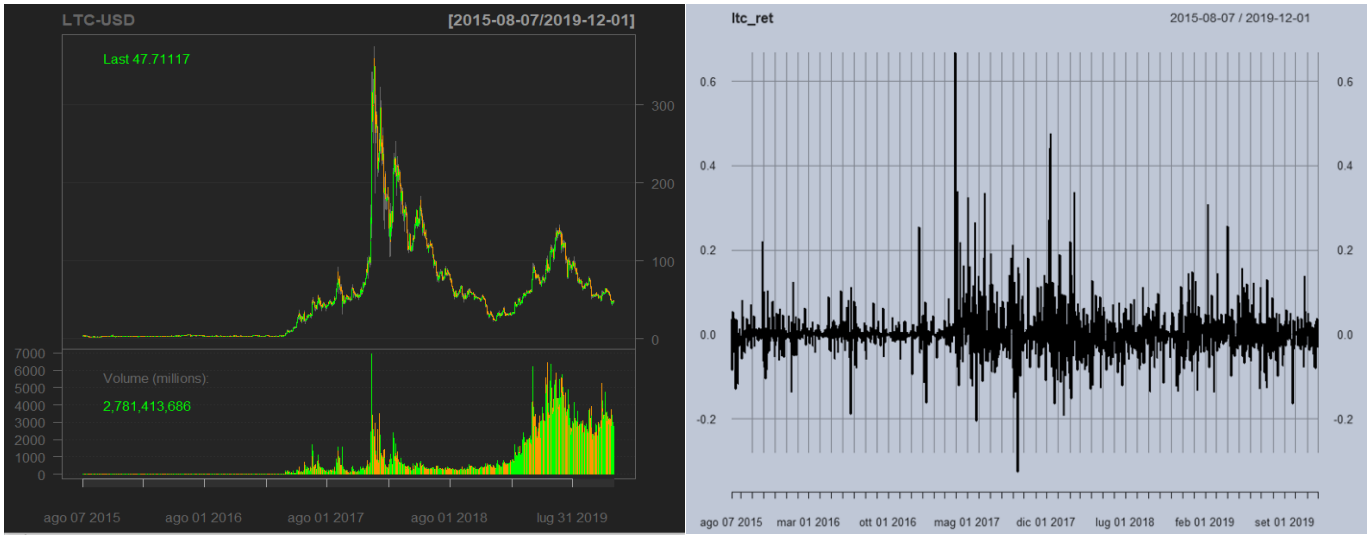

Figura 17: Serie prezzi e rendimenti Litecoin

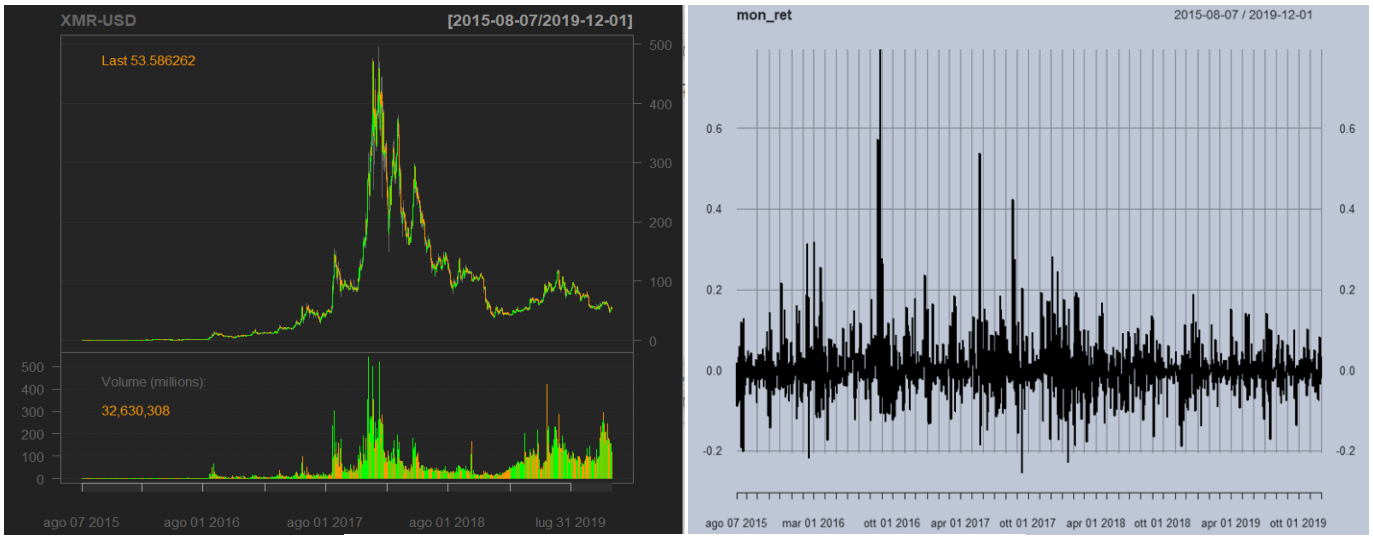

Figura 18: Serie prezzi e rendimenti Monero

	eth_ret		ltc_ret		mon_ret
nobs	1578.000000	nobs	1578.000000	nobs	1578.000000
NAs	0.000000	NAs	0.000000	NAs	0.000000
Minimum	-0.728249	Minimum	-0.326343	Minimum	-0.254101
Maximum	0.507323	Maximum	0.667653	Maximum	0.794167
1. Quartile	-0.024412	1. Quartile	-0.019225	1. Quartile	-0.026665
3. Quartile	0.028268	3. Quartile	0.019688	3. Quartile	0.032420
Mean	0.004969	Mean	0.003153	Mean	0.004974
Median	-0.000790	Median	-0.000937	Median	-0.000748
Sum	7.841148	Sum	4.975816	Sum	7.848228
SE Mean	0.001720	SE Mean	0.001469	SE Mean	0.001748
LCL Mean	0.001595	LCL Mean	0.000272	LCL Mean	0.001545
UCL Mean	0.008343	UCL Mean	0.006035	UCL Mean	0.008402
Variance	0.004669	Variance	0.003406	Variance	0.004822
Stdev	0.068328	Stdev	0.058358	Stdev	0.069438
Skewness	0.287448	Skewness	2.338907	Skewness	2.089263
Kurtosis	14.346994	Kurtosis	19.921057	Kurtosis	17.387702

Figura 19: Statistiche descrittive
Ethereum, Litecoin e Monero

```
Coefficient(s):
          Estimate  Std. Error  t value Pr(>|t|)
btc_ret  0.0028855   0.0009938    2.903  0.00369 **
eth_ret  0.0047591   0.0017227    2.763  0.00573 **
ltc_ret  0.0030591   0.0014737    2.076  0.03792 *
mon_ret  0.0050016   0.0017536    2.852  0.00434 **
btc_ret  0.0304156   0.0336999    0.903  0.36677
eth_ret -0.0616110   0.0584310   -1.054  0.29169
ltc_ret -0.0197015   0.0499609   -0.394  0.69333
mon_ret -0.0210381   0.0594465   -0.354  0.72341
btc_ret -0.0278317   0.0166297   -1.674  0.09421 .
eth_ret  0.0810845   0.0288211    2.813  0.00490 **
ltc_ret -0.0307333   0.0246600   -1.246  0.21266
mon_ret  0.0405496   0.0293422    1.382  0.16699
btc_ret -0.0091480   0.0217570   -0.420  0.67415
eth_ret -0.0034898   0.0376033   -0.093  0.92606
ltc_ret  0.0480751   0.0322503    1.491  0.13604
mon_ret -0.0205957   0.0383762   -0.537  0.59149
btc_ret  0.0096209   0.0168985    0.569  0.56913
eth_ret  0.0021832   0.0291861    0.075  0.94037
ltc_ret  0.0263160   0.0250586    1.050  0.29364
mon_ret -0.0226572   0.0298150   -0.760  0.44730
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
---
Estimates in matrix form:
Constant term:
Estimates:  0.002885472 0.004759097 0.003059117 0.00500163
AR coefficient matrix
AR( 1 )-matrix
        [,1]     [,2]    [,3]    [,4]
[1,]  0.03042 -0.06161 -0.0197 -0.0210
[2,] -0.02783  0.08108 -0.0307  0.0405
[3,] -0.00915 -0.00349  0.0481 -0.0206
[4,]  0.00962  0.00218  0.0263 -0.0227

Residuals cov-matrix:
            [,1]        [,2]        [,3]        [,4]
[1,] 0.001560437 0.001143455 0.001363557 0.001338415
[2,] 0.001143455 0.004649196 0.001595512 0.001776704
[3,] 0.001363557 0.001595512 0.003405644 0.001705010
[4,] 0.001338415 0.001776704 0.001705010 0.004828845
----
aic=  -23.83691
bic=  -23.76893
```

Figura 20: Modello VAR (1) stimato

2.6 Modello DCC

La stima del modello DCC avviene in due passaggi:

- Si stimano *m* modelli GARCH univariati (possono essere anche diversi modelli per ognuna delle serie storiche) per ognuna delle serie storiche considerate (i modelli GARCH sono applicati ai residui ricavati dal modello VAR (1), come nel caso univariato in cui abbiamo utilizzato i residui del modello ARIMA (3, 0, 3)) e vengono così costruite le matrici D_t (vd. par. 1.5.6) dalle quali si ricavano i residui standardizzati.
- Il secondo passaggio è quello appunto di calcolare le correlazioni dinamiche dei residui standardizzati ricavati in precedenza.

Anche in questa sezione utilizziamo diversi modelli GARCH quali SGARCH, EGARCH, IGARCH, GJR-GARCH e TGARCH (tutti i modelli sono abbinati ad una distribuzione t di student). Sulla base dei criteri AIC, BIC, Shibata e Hannan-Quinn (vd. Tab. 3), il modello migliore dovrebbe essere il TGARCH. Tuttavia, scegliamo il modello EGARCH in quanto, a differenza del TGARCH, presenta molti parametri statisticamente significativi.

I parametri stimati nel modello DCC-GARCH sono 33 di cui 24 provengono dalla stima dei GARCH univariati (EGARCH in questo caso) sulle singole serie temporali nella prima fase. Tutti i parametri sono statisticamente significativi, come accennato in precedenza, tranne due (alpha e gamma della serie Ethereum). Inoltre, è importante ricordare che, utilizzando un modello EGARCH, non è necessario porre dei vincoli sui parametri. Infatti, dal momento che viene utilizzata una funzione logaritmica, anche se i parametri sono negativi, la varianza sarà sempre positiva (cfr. par. 1.4.3). Per quanto riguarda il Bitcoin, sia α che β sono molto significativi. Il parametro α (coefficiente ARCH) è maggiore di 0,1 (0,13) e questo implica che la volatilità è sensibile agli eventi di mercato. Il coefficiente GARCH (β), invece, misura la persistenza della volatilità il cui valore è molto alto (0,96). Ciò implica che la volatilità impiega molto tempo a decadere (vd. par. 1.3). Il parametro γ misura l'asimmetria o effetto leverage[19]. Gamma assume un valore vicino a zero (0,04) ma positivo. Essendo statisticamente significativo, ciò implica presenza di asimmetria, ovvero in caso di shocks positivi l'effetto sulla volatilità è più destabilizzante che in caso di shocks negativi. Il parametro α della serie Ethereum è statisticamente non significativo. Mentre il parametro β assume un valore alto (0,88) ed è statisticamente significativo. Quindi, anche in questo caso la volatilità è molto persistente e quindi impiega molto tempo a decadere. Gamma assume un valore positivo (0.22) ma non statisticamente significativo. In merito ai parametri della serie Litecoin, α e β sono statisticamente significativi e assumono rispettivamente i valori 0,14 e 0,989 e quindi possiamo concludere allo stesso modo, ovvero che la volatilità è sensibile agli

[19] L'idea è che, se $\gamma<0$, shocks negativi hanno un impatto maggiore sulla volatilità rispetto a shocks positivi. Se $\gamma>0$, shocks positivi hanno un impatto maggiore sulla volatilità rispetto a shocks negativi. Nel caso multivariato entrambe varianza e covarianza possono reagire differentemente in caso di shocks positivi e negativi.

eventi di mercato ed è molto persistente e quindi impiega tempo a decadere. Invece, il parametro γ è negativo (-0,1) e statisticamente significativo. Ciò implica che shocks negativi hanno maggiore impatto sulla volatilità rispetto a shocks positivi. Nell'ultima serie (Monero), osserviamo un parametro α molto alto (ca. 0,57) quindi la volatilità, in questo caso, è altamente sensibile ad eventi di mercato. Allo stesso tempo, anche β assume un valore elevato 0,89. Il parametro γ assume un valore negativo (ca. -0,36) significativo e
quindi viene riscontrata una forte asimmetria e quindi effetto leverage. Infine, i parametri α e β del DCC sono entrambi molto significativi.

Modello GARCH	AIC	BIC	Shibata	Hannan-Quinn
SGARCH	-36,269	-36,170	-36,269	-36,232
EGARCH	-36,092	-35,980	-36,093	-36,050
IGARCH	-36,217	-36,132	-36,218	-36,186
GJR-GARCH	-36,194	-36,081	-36,195	-36,152
TGARCH	-36,356	-36,243	-36,356	-36,314

Tabella 3: Criteri di valutazione per la scelta del modello

2.6.1 Matrice covarianza/varianza

Quando si lavora con N assets, la varianza dei rendimenti di tali assets risulta essere la matrice del secondo momento, ovvero la matrice delle varianze e covarianze. Tutte le varianze sono raccolte lungo la diagonale di tale matrice mentre le covarianze sono al di fuori di essa. Inoltre, dato che $Cov\left[R_t^i, R_t^j\right] = Cov\left[R_t^j, R_t^i\right]$ (dove R_t^i, R_t^j sono rispettivamente i rendimenti dell'asset$_i$ e dell'asset$_j$ al tempo t) per semplici proprietà delle aspettative, $Cov[R_t]$ è per costruzione una matrice simmetrica. Ad esempio, ogni scelta di portafoglio è basata sulla conoscenza o comunque sulla stima della matrice delle covarianze, in quanto al fine dell'ottimizzazione del portafoglio è importante considerare anche i co-movimenti degli assets presi in coppia. Comunque, come discusso nel capitolo 1, il problema principale di tutti i modelli GARCH multivariati è quello di costruire una matrice delle covarianze e varianze H_t che sia definita positiva in modo tale da garantire che la varianza non sia mai negativa. La matrice H_t stimata dal modello risulta avere tre dimensioni (4 x 4 x 1577), il che significa che abbiamo 1577 matrici 4x4, ovvero una matrice 4x4 per ogni *t*.

2.6.2 Matrice di correlazione

Le correlazioni sono dei fattori critici per molte attività in finanza. Ad esempio, operazioni di hedging (operazioni di copertura) richiedono la stima della correlazione tra i rendimenti degli assets. Se la correlazione e la volatilità cambiano, è necessario un aggiustamento del rapporto di copertura (hedge ratio) per tenere in considerazione le informazioni recenti. Attività come allocazione di titoli e valutazione dei rischi allo stesso

modo necessitano della stima delle correlazioni. La matrice R_t delle correlazioni (cfr. par. 1.5.6) ha allo stesso modo tre dimensioni (4 x 4 x 1577), ovvero 1577 matrici 4 x 4. In figura 21 e 22
mostriamo i grafici delle correlazioni delle quattro criptovalute prese in coppia. Considerando tutto il periodo 2015-2019, nei 6 casi considerati (Bitcoin-Ethereum, Bitcoin-Litecoin, Ethereum-Litecoin, Bitcoin-Monero, Ethereum-Monero, Litecoin-Monero) la correlazione parte da valore intorno allo 0,2 e cresce (in maniera più o meno travagliata) fino ad arrivare ad un valore di 0,8. Le correlazioni, come prevedibile, toccano quasi sempre valori positivi e questo non permette una diversificazione del portafoglio. Per quanto riguarda il primo grafico (Bitcoin-Ethereum), nel periodo agosto 2015-aprile 2017, la correlazione tocca valori in un range tra -0,03-0,63 con un massimo e minimo che vengono raggiunti rispettivamente in agosto 2016 e aprile 2016. Nel periodo successivo maggio 2017-dicembre 2019, la correlazione è più elevata partendo da un minimo di 0,2 fino a toccare valori molto vicini all'unità (ca. 0,98). Nel secondo grafico (Bitcoin-Litecoin), nel periodo agosto 2015-aprile 2017, la correlazione si muove in un range di valori più ampio 0,07-0,93, il cui massimo e minimo viene raggiunto rispettivamente nel periodo gennaio 2016 e ottobre 2015. Nel periodo successivo maggio 2017-dicembre 2019, la correlazione assume sempre valori positivi partendo da valori molto piccoli (ca. 0,003) fino ad arrivare a valori vicino ad uno (ca. 0,97). Nel terzo grafico (Ethereum-Litecoin), nel periodo agosto 2015-aprile 2017, la correlazione tocca valori più bassi in un range -0,16-0,53, nel quale massimo e minimo sono raggiunti rispettivamente in giugno 2016 e dicembre 2016. Nel secondo periodo maggio 2017-dicembre 2019, la correlazione assume valori sempre positivi raggiungendo massimi vicino all'unità. Nel quarto grafico (Bitcoin-Monero), nel periodo agosto 2015-aprile 2017, la correlazione tocca quasi sempre valori in un range 0,3-0,6. Tuttavia, nel periodo agosto 2016-dicembre 2016 c'è un forte calo la cui correlazione arriva a toccare anche minimi negativi (ca. -0.0015). Nel secondo periodo maggio 2017-dicembre 2019, la correlazione assume sempre valori positivi fino a raggiungere massimi vicino all'unità (ca. 0,98). Nel quinto grafico (Ethereum-Monero), nel periodo agosto 2015-aprile 2017, la correlazione gira intorno a valori perlopiù bassi, mostrando anche minimi
negativi (ca. -0,05), tranne che nell'estate 2016 (giugno-agosto) in cui raggiunge valori più elevati in un range tra 0,4-0,6. Nel periodo successivo, maggio 2017-dicembre 2019, vediamo ancora una volta una crescita della correlazione che raggiunge valori molto elevati. Nell'ultimo grafico (Litecoin-Monero), nel periodo agosto 2015-aprile 2017, la correlazione assume valori molto bassi (anche negativi) con dei picchi in gennaio 2016. Nel periodo successivo maggio 2017-dicembre 2019, la correlazione ha una crescita esponenziale, partendo da valori negativi ad arrivare a valori vicino all'unità. Dai grafici è possibile notare come dalla fine dell'anno 2017 e inizio 2018, le correlazioni tra le criptovalute hanno assunto un trend crescente assumendo valori sempre più elevati.

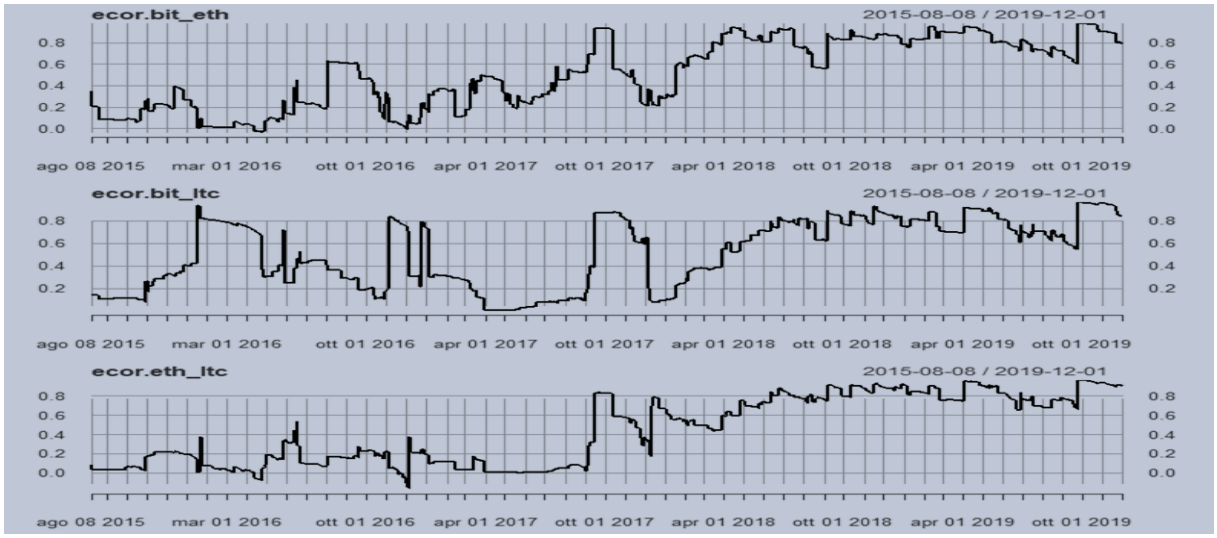

Figura 21: Grafici correlazioni delle criptovalute prese in coppia (Bitcoin-Ethereum, Bitcoin-Litecoin, Ethereum-Litecoin)

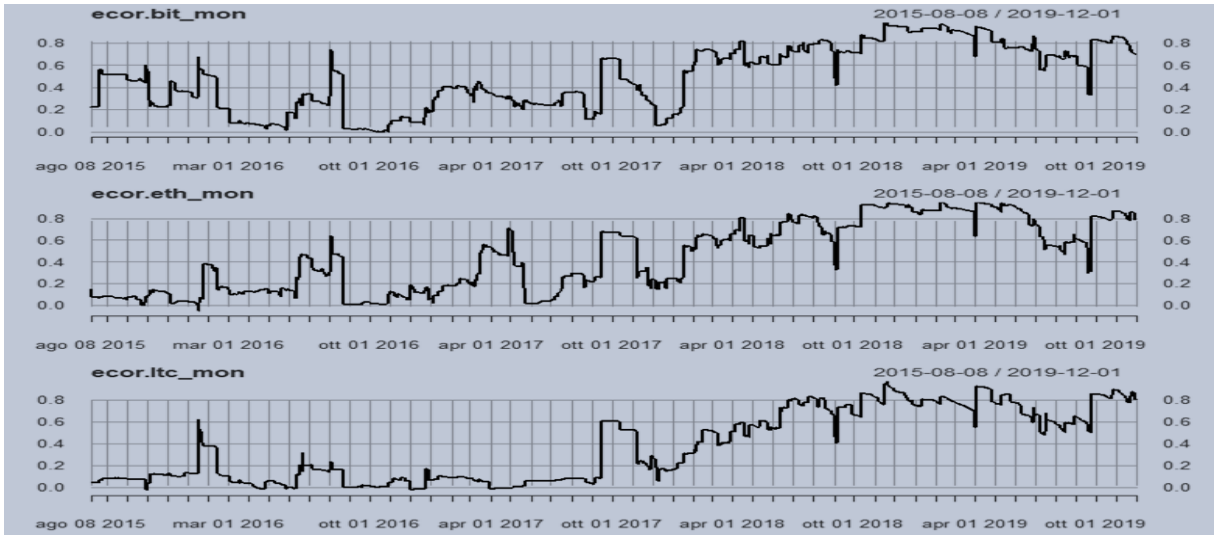

Figura 22: Grafici correlazioni delle criptovalute prese in coppia (Bitcoin-Monero, Ethereum-Monero, Litecoin-Monero)

3 Ottimizzazione portafoglio criptovalute con modelli GARCH

Un portafoglio di investimento è un investimento passivo in titoli, per cui non è necessaria una gestione attiva dei titoli da parte dell'investitore. Può essere anche visto come un investimento fatto da un investitore, il quale non è particolarmente interessato nella gestione della società in cui sta investendo. Quindi, il principale scopo di un investimento in un portafoglio di titoli è il guadagno finanziario che ne deriva. Ogni investitore vorrebbe ottenere il maggior profitto possibile da un investimento, ma questo è controbilanciato da un certo ammontare di rischio che gli investitori sono capaci o comunque sono disposti a prendere con sé. Il rendimento atteso e il rischio misurato dalla varianza (o dalla standard deviation, ovvero la radice quadrata della varianza) sono le due caratteristiche principali di un portafoglio di investimento. Purtroppo, degli studi mostrano come gli investimenti che generano dei rendimenti maggiori siano anche quelli più rischiosi. In questa sezione

utilizzeremo la frontiera efficiente per indicare il portafoglio ottimale. La frontiera efficiente è l'insieme dei portafogli ottimali che offrono il rendimento atteso più elevato per un determinato livello di rischio o il rischio più basso per un determinato livello di rendimento atteso. I portafogli che si trovano al di sotto della frontiera efficiente non sono ottimali perché non forniscono un rendimento sufficiente per il livello di rischio. I portafogli che raggruppano a destra della frontiera efficiente sono non ottimali perché presentano un livello più elevato di rischio per il tasso di rendimento definito. La frontiera efficiente fu introdotta dal premio Nobel per l'economia Harry Markowitz nel 1952. Per la costruzione della frontiera efficiente faremo uso delle stime dei rendimenti attesi e delle varianze e covarianze attese del modello DCC utilizzato nel precedente capitolo.

3.1 Rendimenti attesi e varianze e covarianze attese

In questo paragrafo analizzeremo il *forecast* (con un orizzonte temporale di 300 periodi) del modello DCC stimato nel par. 2.6. Dalla figura 23 mostriamo il forecast delle prime e delle ultime due matrici di correlazione. Dalla figura 24 e 25, invece, mostriamo i grafici dei forecast delle correlazioni delle criptovalute prese a due a due (Bitcoin-Ethereum, Bitcoin-Litecoin, Ethereum-Litecoin Bitcoin-Monero, Ethereum-Bitcoin, Litecoin-Monero). I 6 grafici hanno un andamento decrescente. Il modello, quindi, prevede una riduzione delle correlazioni nei 300 periodi futuri. Tuttavia, ciò che interessa a noi per il calcolo della frontiera efficiente sono i rendimenti attesi e le varianze e covarianze attese. I forecast dei rendimenti e delle varianze/covarianze sono oggetti di dimensioni, rispettivamente, (300 x 4) e (4 x 4 x 300). Di questi, calcoliamo i valori medi e li annualizziamo in quanto valori giornalieri. In tabella 4 e 5 mostriamo i risultati.

```
*-----------------------------------*
*       DCC GARCH Forecast          *
*-----------------------------------*

Distribution         :  mvt
Model                :  DCC(1,1)
Horizon              :  300
Roll Steps           :  0
-----------------------------------

0-roll forecast:

First 2 Correlation Forecasts
, , 1

       [,1]   [,2]   [,3]   [,4]
[1,] 1.0000 0.7881 0.8391 0.6933
[2,] 0.7881 1.0000 0.9016 0.7891
[3,] 0.8391 0.9016 1.0000 0.8027
[4,] 0.6933 0.7891 0.8027 1.0000

, , 2

       [,1]   [,2]   [,3]   [,4]
[1,] 1.0000 0.7881 0.8391 0.6933
[2,] 0.7881 1.0000 0.9015 0.7891
[3,] 0.8391 0.9015 1.0000 0.8027
[4,] 0.6933 0.7891 0.8027 1.0000

. . . .
. . . .

Last 2 Correlation Forecasts
, , 1

       [,1]   [,2]   [,3]   [,4]
[1,] 1.0000 0.7838 0.8324 0.6887
[2,] 0.7838 1.0000 0.8937 0.7828
[3,] 0.8324 0.8937 1.0000 0.7954
[4,] 0.6887 0.7828 0.7954 1.0000

, , 2

       [,1]   [,2]   [,3]   [,4]
[1,] 1.0000 0.7838 0.8324 0.6887
[2,] 0.7838 1.0000 0.8937 0.7828
[3,] 0.8324 0.8937 1.0000 0.7954
[4,] 0.6887 0.7828 0.7954 1.0000
```

Figura 23: Output forecast modello DCC (300 periodi)

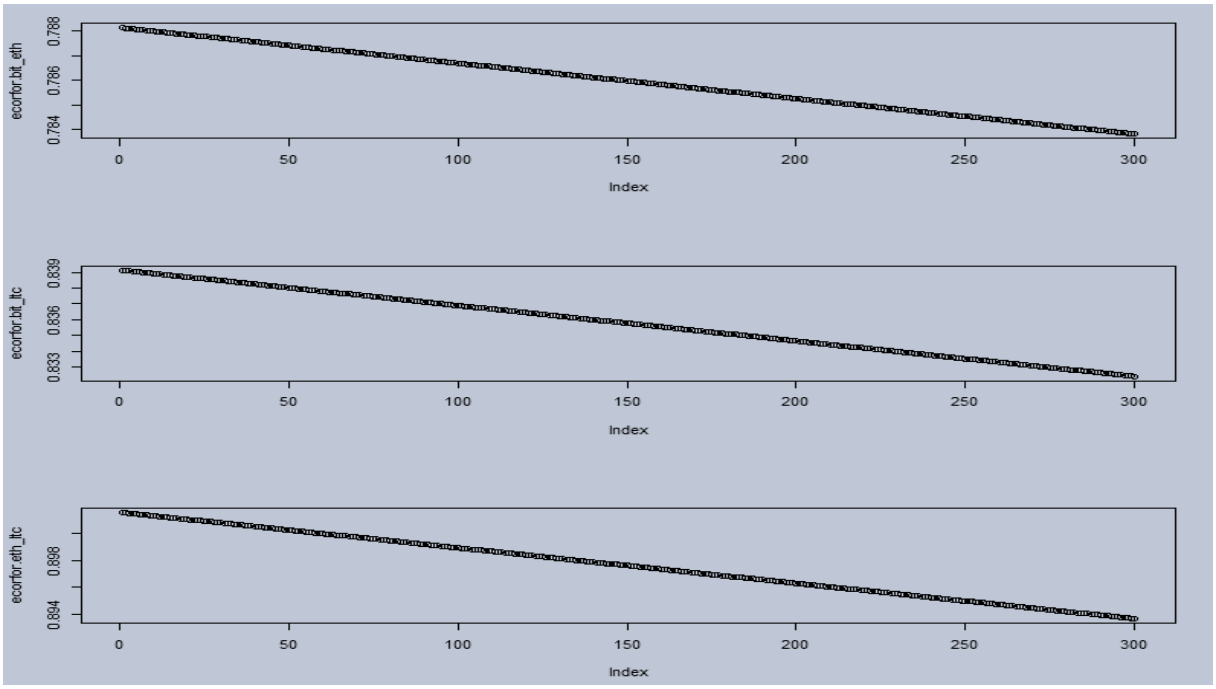

Figura 24: Grafici forecast correlazioni a due a due delle criptovalute (Bitcoin-Ethereum, Bitcoin-Litecoin, Ethereum-Litecoin)

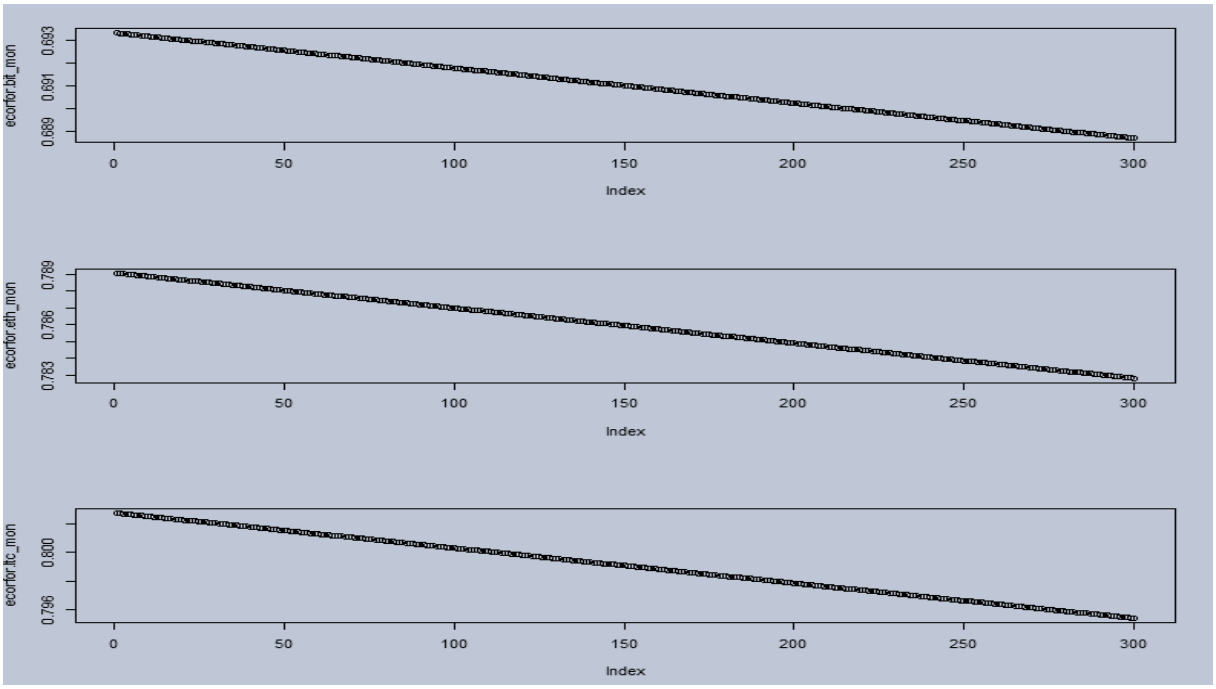

Figura 25: Grafici forecast correlazioni a due a due delle criptovalute (Ethereum-Monero, Litecoin-Monero, Litecoin-Monero)

3.2 Frontiera efficiente

In questa sezione andremo a calcolare la nostra frontiera efficiente. La frontiera efficiente sarà calcolata sulla base dei momenti ricavati dal modello DCC nel precedente paragrafo. Al fine dell'ottimizzazione del portafoglio, andremo a minimizzare il rischio (con l'aiuto di R il cui codice è in appendice). La misura di rischio utilizzata nell'ottimizzazione è *mean-variance*. L'analisi della *mean-variance* è il processo di

Asset	μ
Bitcoin	0,03158611
Ethereum	0,14913142
Litecoin	0,05132435
Monero	0,17827976

Tabella 4: Rendimenti attesi

Asset	Bitcoin	Ethereum	Litecoin	Monero
Bitcoin	6,808410e-06	2,571737e-05	6,463588e-06	2,015244e-05
Ethereum	2,571737e-05	1,692086e-04	3,020716e-05	1,160022e-04
Litecoin	6,463588e-06	3,020716e-05	1,021474e-05	2,453398e-05
Monero	2,015244e-05	1,160022e-04	2,453398e-05	1,297008e-04

Tabella 5: Matrice covarianza/varianza attesa

ponderazione del rischio, espresso come varianza, rispetto al rendimento atteso. Gli investitori utilizzano l'analisi della media varianza per decidere in quali strumenti finanziari investire, sulla base di quanto rischio sono disposti ad assumersi in cambio di diversi livelli di remunerazione. L'analisi della *mean-variance* consente agli investitori di trovare la remunerazione più elevata a un determinato livello di rischio o il minimo rischio a un determinato livello di rendimento. L'analisi della *mean-variance* è una parte della *modern portfolio theory* di Harry Markowitz, che presuppone che gli investitori prendano decisioni razionali sugli investimenti se dispongono di informazioni complete. Un'assunzione è che gli investitori vogliono basso rischio e alto rendimento. Dalla figura 26 vediamo le specifiche dell'ottimizzazione del nostro portafoglio (PARMA è il pacchetto utilizzato su R, ovvero portfolio allocation and risk management applications). Il problema di minimizzazione del rischio verrà risolto con due metodi: QP (quadratic programming) e SOCP (Second Order Cone Programming). Inoltre, imponiamo dei vincoli sui pesi:

- Peso minimo $\geq 0{,}02$
- Peso massimo $\leq 0{,}5$

```
+---------------------------------+
|       PARMA Specification       |
+---------------------------------+
No.Assets               : 4
Problem                 : QP,SOCP
Input                   : Covariance
Risk Measure    : EV
Objective               : minrisk
```

Figura 26: Specificazione ottimizzazione portafoglio

Dalle tabelle 6 e 7 possiamo vedere le soluzioni derivanti dalle due modalità QP e SOCP, ovvero i pesi ottimali e il rendimento e il rischio atteso. I risultati ottenuti con le due diverse modalità sono molto simili. Quindi, la scelta dell'uno o dell'altro è indifferente. Infine, andiamo a tracciare la nostra frontiera efficiente. In figura 27 osserviamo la frontiera efficiente. Il portafoglio ottimale è calcolato come rapporto tra rischio atteso e rendimento atteso (i cui valori sono in tabella 7). Le due frontiere efficienti calcolate con le due diverse modalità coincidono, come era già possibile vedere dalle tabelle 6 e 7. Vediamo, invece, cosa succede nel caso in cui non ci sono vincoli sui pesi dei singoli asset. Nelle tabelle 8 e 9 osserviamo le soluzioni. Mentre i pesi cambiano (addirittura 0 per il Bitcoin e l'Ethereum nel caso QP), il rendimento e il rischio atteso sono piuttosto simili al caso con vincoli (il rischio atteso è leggermente inferiore). In figura 28 osserviamo la frontiera efficiente. Anche in questo caso le due frontiere efficienti calcolate con le due diverse modalità coincidono. Tuttavia, il portafoglio ottimale in figura 28 è preferibile a quello ottenuto nel caso di ottimizzazione con vincoli sui pesi. Ciò nonostante, senza l'utilizzo di vincoli è possibile che vengano prodotti dei portafogli troppo concentrati e questo è molto rischioso. Infatti, dato che non c'è certezza che il modello e le stime siano esatte, la diversificazione è una contromisura per difendersi da eventi non prevedibili.

Asset	Peso (QP)	Peso (SOCP)
Bitcoin	0.07934002	0.07933486
Ethereum	0.02	0.02
Litecoin	0.5	0.5
Monero	0.40065998	0.40065514

Tabella 6: Pesi ottimali (con vincoli)

	QP	SOCP
Rendimento atteso	0.102580409137952	0.102579383333861
Rischio atteso	3.76535878291502e-05	3.76528006032586e-05

Tabella 7: Rendimento e rischio atteso (con vincoli)

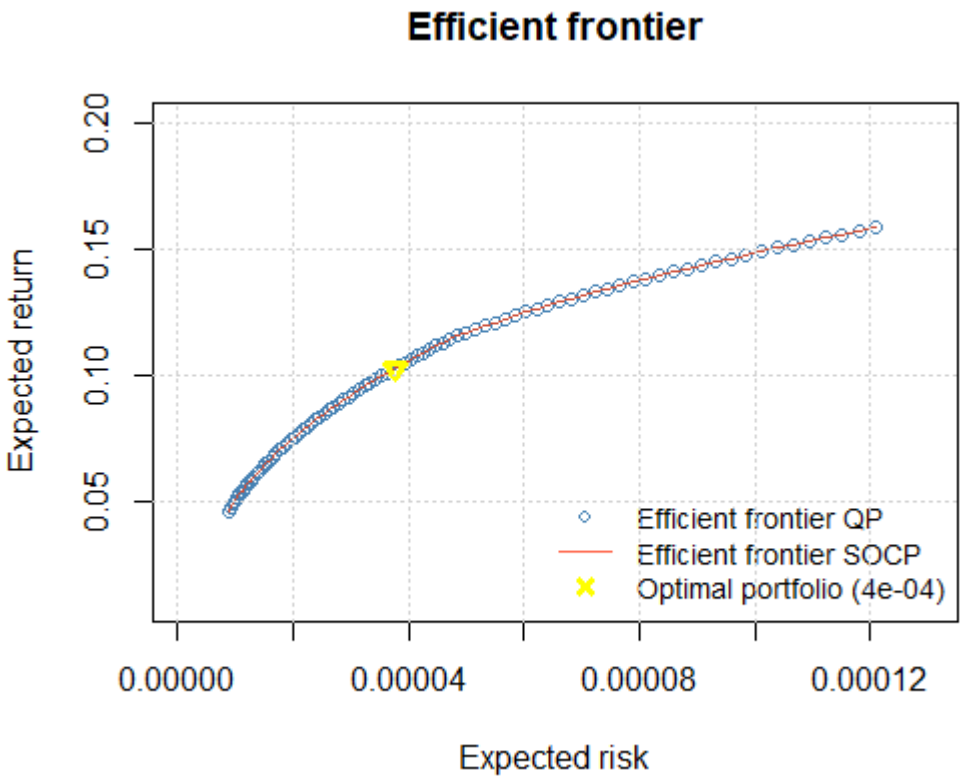

Figura 27: Frontiera efficiente con vincoli

Asset	Peso (QP)	Peso (SOCP)
Bitcoin	0	5.041936e-15
Ethereum	0	4.413864e-13
Litecoin	0.5962672	5.962613e-01
Monero	0.4037328	4.037287e-01

Tabella 8: Pesi ottimali (senza vincoli)

	QP	SOCP
Rendimento atteso	0.102580409137952	0.102579383333861
Rischio atteso	3.65852127481457e-05	3.65844810475568e-05

Tabella 9: Rendimento e rischio atteso (senza vincoli)

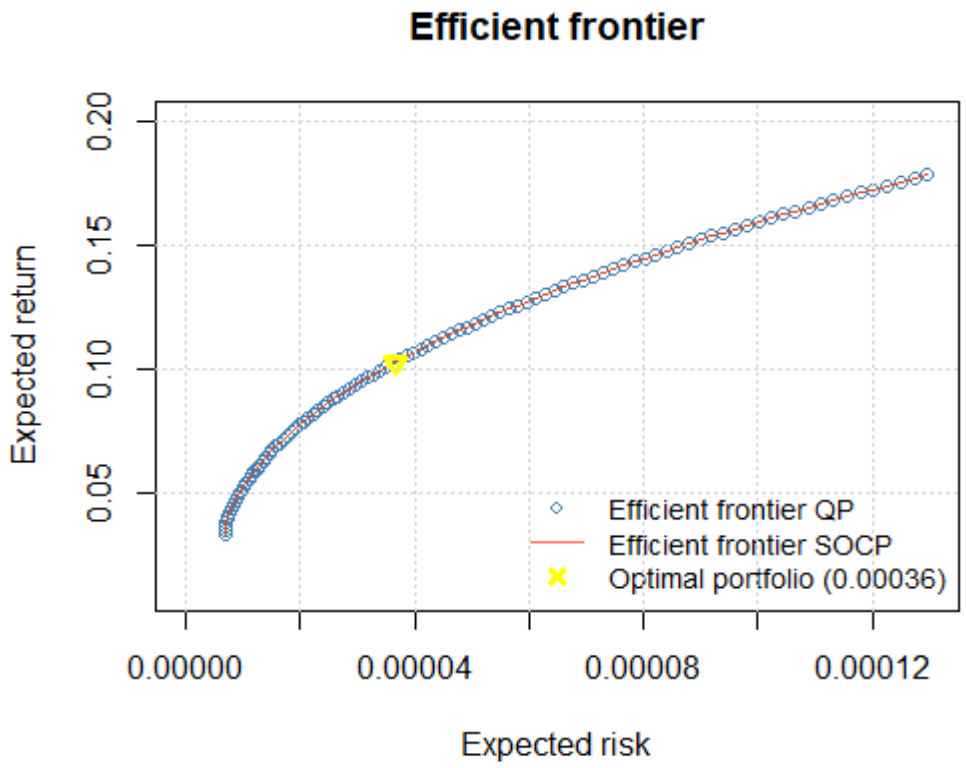

Figura 28: Frontiera efficiente senza vincoli

3.3 Value at risk

Le misure del rischio danno importanti informazioni ai managers che appunto si occupano di gestire i rischi di istituti finanziari. Il value at risk (VaR) è un metodo statistico che calcola un singolo numero per riassumere il rischio complessivo di un portafoglio finanziario. Potrebbe essere anche utilizzato per definire il capitale che una banca deve mantenere rispetto ai rischi che sta assumendo. Quando si utilizza il VaR, si sta affermando che: "Sono X per cento sicuro che l'istituto finanziario non incorrerà una perdita maggiore di V nei prossimi T giorni". La variabile V indica il quantile del VaR del portafoglio. Il value at risk è una funzione con due parametri: il livello di confidenza (X%) e l'orizzonte temporale o il periodo di detenzione. Il VaR potrebbe anche essere definito come il livello di perdita in un periodo di T giorni che ha una probabilità (100-X) % di essere superato. Nelle banche, le autorità di regolamentazione richiedono il calcolo del VaR per i rischi di mercato con un orizzonte temporale di 10 giorni e un livello di confidenza di X = 99 %. Se prendiamo T come periodo di detenzione e X% come livello di confidenza, il VaR è la perdita che corrisponde al (100-X) percentile della distribuzione dei guadagni del portafoglio nei prossimi T giorni. Inoltre, interessante da accennare, se si tiene conto della distribuzione di probabilità del guadagno, il value at risk è relativo alla coda sinistra della distribuzione mentre se consideriamo la distribuzione delle perdite, il value at risk è relativo alla coda destra della distribuzione. Nel primo caso, la perdita è un guadagno negativo, mentre nel secondo il guadagno è una perdita negativa (figura 29). Esistono diversi metodi per la stima del VaR, ma qui utilizzeremo *Historical simulation* e l'utilizzo di GARCH univariati sulla serie dei rendimenti del portafoglio.

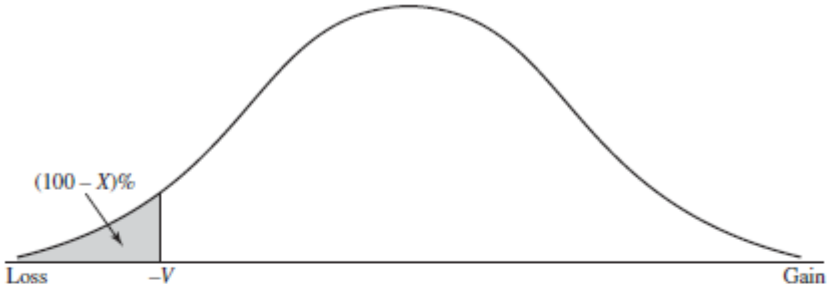

FIGURE 12.1 Calculation of VaR from the Probability Distribution of the Gain in the Portfolio Value
Losses are negative gains; confidence level is X%; VaR level is V.

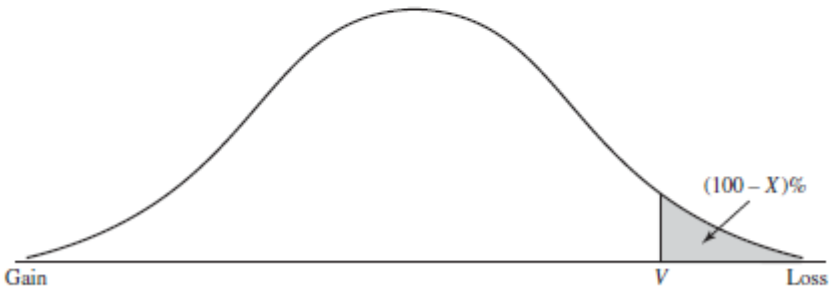

FIGURE 12.2 Calculation of VaR from the Probability Distribution of the Loss in the Portfolio Value
Gains are negative losses; confidence level is X%; VaR level is V.

Figura 29: Distribuzione di probabilità dei guadagni e delle perdite. Fonte: John C. Hull, Risk management and financial institutions (quarta edizione), p. 257, Wiley (2015).

3.4 Historical simulation

Come il nome suggerisce, *historical simulation* è un metodo che usa osservazioni passate dei rendimenti per effettuare previsioni del VaR. Quindi, l'idea principale è quella di prevedere delle future perdite sulla base delle performance passate. Il primo step è quello di calcolare i rendimenti del portafoglio considerando i pesi mostrati in tabella 6 (indifferente tra QP o SOCP, dato che sono quasi uguali). I rendimenti vanno ordinati in ordine crescente (quindi dal valore più piccolo a quello più grande). Successivamente, va trovato il p% quantile della serie ordinata. Ad esempio, se $p = 0{,}01$ (in caso di VaR (99%)) allora $q = p * N$ dove N è il numero delle osservazioni. Nel nostro caso, $q = 0{,}01 * 1577 = 15{,}77 \sim 16$. Infine, estraiamo il q-esimo valore dalla serie ordinata. Quindi, nel nostro caso, estrarremmo il sedicesimo valore. Il VaR sarà il valore estratto dalla serie ordinata dei rendimenti. Questo metodo di stima del VaR ha alcuni vantaggi e svantaggi. I vantaggi sono (Danielsson, 2011):

- non è necessario assumere o identificare una distribuzione per i rendimenti
- questo metodo definisce direttamente le dipendenze non lineari
- le code grasse della distribuzione sono catturate se si è a disposizione di molte osservazioni.

Gli svantaggi sono (Dowd 2002):

- totale dipendenza dalle osservazioni
- sono necessarie molte osservazioni
- non è possibile catturare delle rotture strutturali nella volatilità.

I risultati sono mostrati in tabella 10. I livelli di confidenza utilizzati sono $X = 99\%$ e $X = 95\%$.

Possiamo concludere che:

- siamo confidenti al 99% che il portafoglio non incorrerà in una perdita maggiore del 13,3 %
- siamo confidenti al 95% che il portafoglio non incorrerà in una perdita maggiore dell'8,82%.

Nel peggior scenario le perdite sono molto elevate, questo è dovuto al fatto che la distribuzione dei rendimenti ha delle code molto ampie.

VaR (99%)	-0,133
VaR (95%)	-0,0882

Tabella 10: VaR portafoglio con livello di confidenza X = 99% e X = 95%

3.5 GARCH univariati per la stima del Value at Risk

La stima del VaR in questo caso consiste in tre steps:

- stima della volatilità utilizzando dei modelli GARCH univariati (applicando tali modelli sulla serie dei rendimenti del portafoglio)
- *forecast* della volatilità utilizzando i modelli di cui sopra (TGARCH, GJR-GARCH, EGARCH, SGARCH, IGARCH)
- calcolo del VaR utilizzando la volatilità condizionata predetta.

Ipotizzando $\mu = 0$, il VaR sarà così calcolato:

$$VaR = \sigma_{t+1} * F^{-1}(\alpha) \qquad (51)$$

dove σ_{t+1} è la standard deviation predetta per t+1 condizionata ai valori passati e F^{-1} è la funzione inversa della densità di probabilità della distribuzione t di student. Nella stima dei modelli viene utilizzata la distribuzione t di student. In tabella 11 vengono mostrati i criteri di valutazione dei modelli stimati (AIC, BIC, Shibata, Hannan-Quinn). Sulla base di tali criteri il miglior modello è l'IGARCH. Tutti i modelli stimati non presentano autocorrelazioni seriali nei residui standardizzati e nei residui al quadrato, infatti il Ljung-Box test effettuato sui residui dei vari modelli presenta dei p-values molto elevati (figure 30-34). Ciò significa che la probabilità di osservare residui incorrelati è molto alta. Inoltre, il modello EGARCH è l'unico, tra i modelli asimmetrici utilizzati, che è riesce a catturare l'effetto leverage tramite il parametro gamma, il cui valore è positivo (ca. 0,35) e statisticamente significativo. Quindi, shocks positivi hanno un effetto più destabilizzante rispetto a shocks negativi. I parametri dei modelli stimati sono mostrati nelle figure 35-39. È interessante da notare anche che, secondo l'Adjusted Pearson Goodness-of-Fit Test, i residui standardizzati di tutti i modelli

stimati seguono una distribuzione t di student. Infatti, i p-values ottenuti dal test statistico sono molto elevati e quindi non è possibile rifiutare l'ipotesi nulla in base alla quale i residui standardizzati seguono una distribuzione t di student. Questo è possibile vederlo anche dall'analisi grafica dei Q-Q Plots (figure 40-42). Infatti, i quantili della distribuzione dei rendimenti del portafoglio giacciono lungo la retta che rappresenta la distribuzione t di student. Il prossimo step sarà quello di fare un forecast della volatilità condizionata stimata di un periodo in modo da permetterci di computare il value at risk. In tabella 12 mostriamo i VaR ottenuti, con un livello di confidenza del 95% e del 99%, dai forecast dei diversi modelli stimati. Il VaR è calcolato secondo l'equazione (50), i cui valori critici, al 95% e al 99%, sono: $F^{-1}(0{,}05) \sim -1{,}18$; $F^{-1}(0{,}01) \sim -2{,}44$ (ottenuti su R, vd. Appendice).

Modello GARCH	AIC	BIC	Shibata	Hannan-Quinn
SGARCH	-3,2673	-3,2503	-3,2674	-3,2610
EGARCH	-3,2685	-3,2481	-3,2686	-3,2610
IGARCH	-3,2687	-3,2551	-3,2687	-3,2636
GJR-GARCH	-3,2670	-3,2466	-3,2671	-3,2594
TGARCH	-3,2667	-3,2464	-3,2668	-3,2592

Tabella 11: Criteri valutazione modelli GARCH applicati alla serie dei rendimenti del portafoglio

```
Weighted Ljung-Box Test on Standardized Residuals
------------------------------------
                        statistic p-value
Lag[1]                     0.1943  0.6594
Lag[2*(p+q)+(p+q)-1][2]    0.4104  0.7372
Lag[4*(p+q)+(p+q)-1][5]    3.3360  0.3491
d.o.f=0
H0 : No serial correlation

Weighted Ljung-Box Test on Standardized Squared Residuals
------------------------------------
                        statistic p-value
Lag[1]                     0.2956  0.5866
Lag[2*(p+q)+(p+q)-1][5]    1.6245  0.7093
Lag[4*(p+q)+(p+q)-1][9]    6.4774  0.2470
d.o.f=2
```

Figura 30: Test sui residui standardizzati e residui al quadrato (EGARCH)

```
Weighted Ljung-Box Test on Standardized Residuals
------------------------------------
                        statistic p-value
Lag[1]                     0.2674  0.6051
Lag[2*(p+q)+(p+q)-1][2]    0.4451  0.7199
Lag[4*(p+q)+(p+q)-1][5]    3.2535  0.3627
d.o.f=0
H0 : No serial correlation

Weighted Ljung-Box Test on Standardized Squared Residuals
------------------------------------
                        statistic p-value
Lag[1]                      0.456  0.4995
Lag[2*(p+q)+(p+q)-1][5]     1.430  0.7571
Lag[4*(p+q)+(p+q)-1][9]     6.907  0.2070
d.o.f=2
```

Figura 31: Test sui residui standardizzati e residui al quadrato (GJR-GARCH)

```
Weighted Ljung-Box Test on Standardized Residuals
------------------------------------
                        statistic p-value
Lag[1]                     0.2350  0.6279
Lag[2*(p+q)+(p+q)-1][2]    0.4432  0.7209
Lag[4*(p+q)+(p+q)-1][5]    3.7493  0.2870
d.o.f=0
H0 : No serial correlation

Weighted Ljung-Box Test on Standardized Squared Residuals
------------------------------------
                        statistic p-value
Lag[1]                    0.02928 0.86413
Lag[2*(p+q)+(p+q)-1][5]   1.65679 0.70134
Lag[4*(p+q)+(p+q)-1][9]  10.87517 0.03239
d.o.f=2
```

Figura 32: Testi sui residui standardizzati e sui residui al quadrato (TGARCH)

```
Weighted Ljung-Box Test on Standardized Residuals
------------------------------------
                        statistic p-value
Lag[1]                     0.3733  0.5412
Lag[2*(p+q)+(p+q)-1][2]    0.5767  0.6587
Lag[4*(p+q)+(p+q)-1][5]    3.4853  0.3255
d.o.f=0
H0 : No serial correlation

Weighted Ljung-Box Test on Standardized Squared Residuals
------------------------------------
                        statistic p-value
Lag[1]                     0.3908  0.5319
Lag[2*(p+q)+(p+q)-1][5]    1.2279  0.8063
Lag[4*(p+q)+(p+q)-1][9]    7.3907  0.1686
d.o.f=2
```

Figura 33: Test sui residui standardizzati e sui residui al quadrato (SGARCH)

```
Weighted Ljung-Box Test on Standardized Residuals
------------------------------------
                        statistic p-value
Lag[1]                     0.3775  0.5390
Lag[2*(p+q)+(p+q)-1][2]    0.5811  0.6567
Lag[4*(p+q)+(p+q)-1][5]    3.4901  0.3248
d.o.f=0
H0 : No serial correlation

Weighted Ljung-Box Test on Standardized Squared Residuals
------------------------------------
                        statistic p-value
Lag[1]                     0.3908  0.5319
Lag[2*(p+q)+(p+q)-1][5]    1.2278  0.8063
Lag[4*(p+q)+(p+q)-1][9]    7.3627  0.1707
d.o.f=2
```

Figura 34: Testi sui residui standardizzati e sui residui al quadrato

```
GARCH Model     : eGARCH(1,1)
Mean Model      : ARFIMA(0,0,0)
Distribution    : std

Optimal Parameters
------------------------------------
        Estimate  Std. Error  t value Pr(>|t|)
mu      0.001337    0.000820   1.6307 0.102964
omega  -0.249500    0.080632  -3.0943 0.001973
alpha1  0.045577    0.025995   1.7533 0.079556
beta1   0.957005    0.013829  69.2050 0.000000
gamma1  0.356097    0.049945   7.1298 0.000000
shape   3.134226    0.276599  11.3313 0.000000

Robust Standard Errors:
        Estimate  Std. Error  t value Pr(>|t|)
mu      0.001337    0.000695   1.9240 0.054355
omega  -0.249500    0.087148  -2.8630 0.004197
alpha1  0.045577    0.026419   1.7251 0.084505
beta1   0.957005    0.015275  62.6514 0.000000
gamma1  0.356097    0.046726   7.6209 0.000000
shape   3.134226    0.284814  11.0045 0.000000

LogLikelihood : 2584.878
```

Figura 35: Modello EGARCH

```
GARCH Model     : gjrGARCH(1,1)
Mean Model      : ARFIMA(0,0,0)
Distribution    : std

Optimal Parameters
------------------------------------
        Estimate  Std. Error  t value Pr(>|t|)
mu      0.001116    0.000877   1.2723 0.203260
omega   0.000118    0.000039   3.0277 0.002464
alpha1  0.219472    0.042611   5.1506 0.000000
beta1   0.806789    0.029397  27.4442 0.000000
gamma1 -0.054522    0.044712  -1.2194 0.222689
shape   3.390631    0.273181  12.4117 0.000000

Robust Standard Errors:
        Estimate  Std. Error  t value Pr(>|t|)
mu      0.001116    0.000761   1.4657 0.142733
omega   0.000118    0.000049   2.4360 0.014852
alpha1  0.219472    0.037680   5.8246 0.000000
beta1   0.806789    0.031822  25.3535 0.000000
gamma1 -0.054522    0.045717  -1.1926 0.233031
shape   3.390631    0.262710  12.9064 0.000000

LogLikelihood : 2583.681
```

Figura 36: Modello GJR-GARCH

```
GARCH Model     : fGARCH(1,1)
fGARCH Sub-Model        : TGARCH
Mean Model      : ARFIMA(0,0,0)
Distribution    : std

Optimal Parameters
------------------------------------
        Estimate  Std. Error  t value Pr(>|t|)
mu      0.001432    0.000930   1.5403 0.123483
omega   0.002473    0.000759   3.2593 0.001117
alpha1  0.205086    0.032876   6.2382 0.000000
beta1   0.827031    0.026447  31.2717 0.000000
eta11  -0.158582    0.086077  -1.8423 0.065427
shape   3.133707    0.274251  11.4264 0.000000

Robust Standard Errors:
        Estimate  Std. Error  t value Pr(>|t|)
mu      0.001432    0.000904   1.5849 0.112995
omega   0.002473    0.000824   3.0030 0.002673
alpha1  0.205086    0.031271   6.5583 0.000000
beta1   0.827031    0.026806  30.8530 0.000000
eta11  -0.158582    0.092449  -1.7153 0.086283
shape   3.133707    0.284254  11.0243 0.000000

LogLikelihood : 2583.465
```

Figura 37: Modello TGARCH

```
GARCH Model      : sGARCH(1,1)
Mean Model       : ARFIMA(0,0,0)
Distribution     : std

Optimal Parameters
------------------------------------
        Estimate  Std. Error  t value Pr(>|t|)
mu      0.000935    0.000867   1.0784 0.280871
omega   0.000121    0.000040   3.0622 0.002197
alpha1  0.195144    0.033736   5.7845 0.000000
beta1   0.803856    0.029567  27.1880 0.000000
shape   3.401186    0.271406  12.5317 0.000000

Robust Standard Errors:
        Estimate  Std. Error  t value Pr(>|t|)
mu      0.000935    0.000728   1.2844 0.198990
omega   0.000121    0.000049   2.4697 0.013522
alpha1  0.195144    0.028148   6.9327 0.000000
beta1   0.803856    0.031983  25.1335 0.000000
shape   3.401186    0.261712  12.9959 0.000000

LogLikelihood : 2582.928
```

Figura 38: Modello SGARCH

```
GARCH Model      : iGARCH(1,1)
Mean Model       : ARFIMA(0,0,0)
Distribution     : std

Optimal Parameters
------------------------------------
        Estimate  Std. Error  t value Pr(>|t|)
mu      0.000933    0.000865   1.0781 0.280991
omega   0.000121    0.000037   3.2429 0.001183
alpha1  0.195900    0.028948   6.7674 0.000000
beta1   0.804100          NA       NA       NA
shape   3.393566    0.203973  16.6373 0.000000

Robust Standard Errors:
        Estimate  Std. Error  t value Pr(>|t|)
mu      0.000933    0.000820   1.1376 0.255270
omega   0.000121    0.000042   2.8553 0.004299
alpha1  0.195900    0.031246   6.2697 0.000000
beta1   0.804100          NA       NA       NA
shape   3.393566    0.208736  16.2577 0.000000

LogLikelihood : 2582.979
```

Figura 39: Modello IGARCH

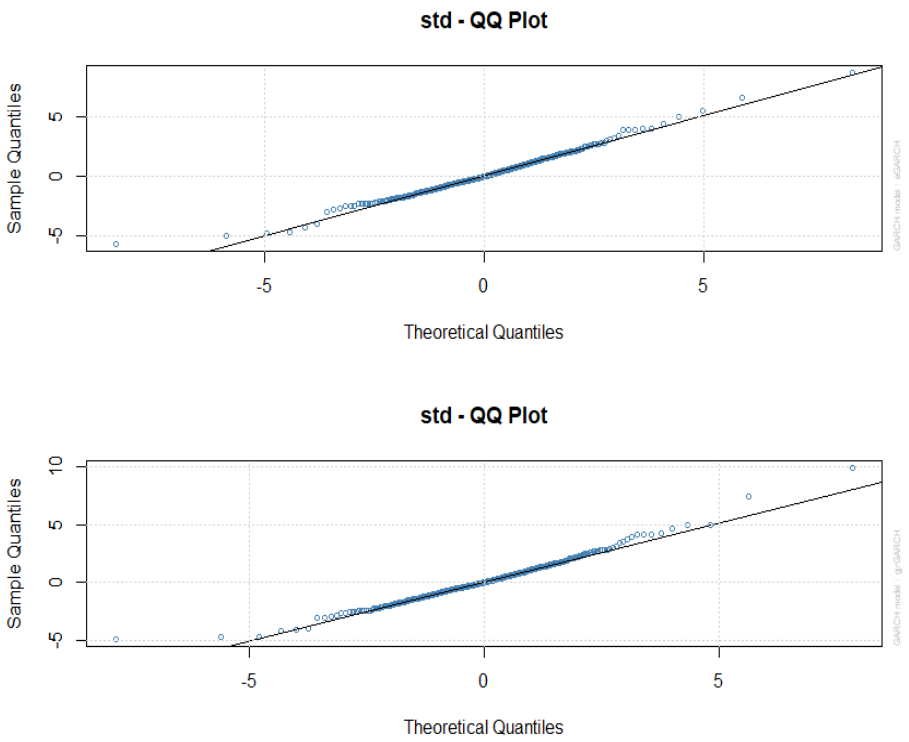

Figura 40: QQ plots, EGARCH (figura sopra) e GJR-GARCH (figura sotto)

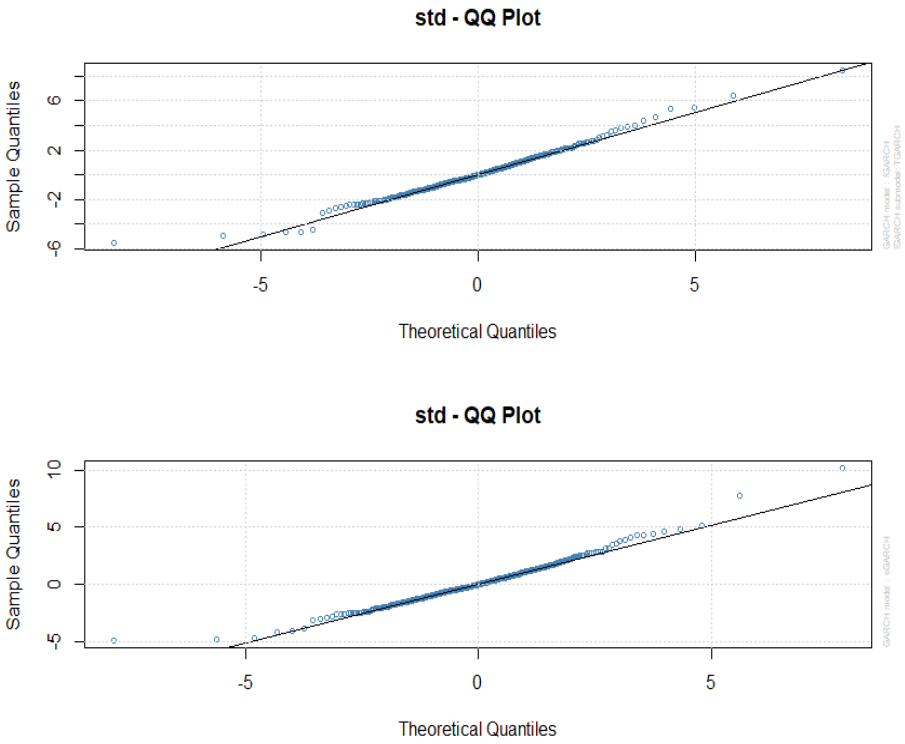

Figura 41: QQ plots, TGARCH (figura sopra) e SGARCH (figura sotto)

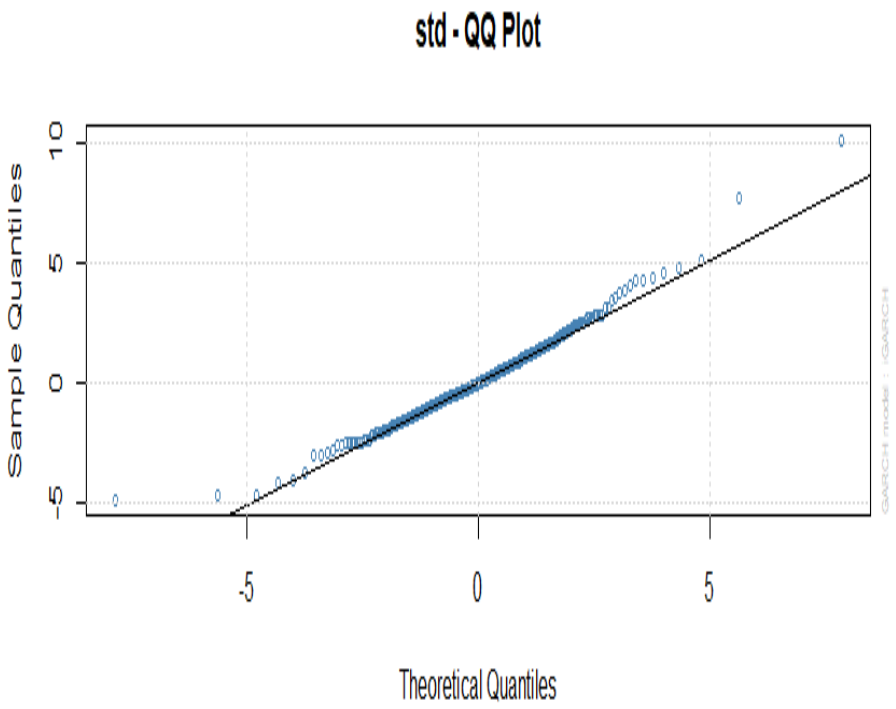

Figura 42: QQ plot, IGARCH

3.6 Test di valutazione

Dopo aver due tecniche nella stima del value at risk, bisogna verificare che la sua accuratezza attraverso diversi test statistici. Ci sono molte metodologie ed è necessario trovare il miglior modello per la stima del rischio. A tal fine, utilizzeremo tre tecniche statistiche per la valutazione della qualità del forecast del VaR:

- Violation ratio
- Il test di Kupiec
- Il test di Christoffersen

Il primo metodo è relativo al confronto retrospettivo dei valori VaR fuori dal campione con osservazioni storiche (backtesting). Gli altri due metodi sono semplicemente dei test statistici.

3.6.1 Violation ratio

Se le perdite effettive eccedono i forecast del VaR, allora il value at risk è considerate violato. Il rapporto di violazione (violation ratio) è la somma dei casi di superamento diviso per il numero atteso di casi di superamento. Il rapporto è quindi dato da (Danielsson, 2011):

$$VR = \frac{E}{p*N} \qquad (52)$$

dove

- E è il numero osservato di casi di superamento
- p è il livello di probabilità del VaR, nel nostro caso 0,05 e 0,01
- N è il numero di osservazioni utilizzate per il forecast del value at risk

Molti risk managers concordano che il valore ottimale di tale rapporto debba trovarsi tra 0,8 e 1,2. Nel caso contrario, il modello è imperfetto. Tale metodo ha il vantaggio di essere facile da utilizzare e quindi può essere utile per una prima veloce analisi. Però ha anche degli svantaggi, come quello di non riuscire a mostrare quali sono le cause del fallimento del modello. In questo caso vengono in aiuto altre metodologie che abbiamo citato sopra: i test statistici di Kupiec e Christoffersen.

3.6.2 Test di Kupiec

Assumiamo che i casi di superamente nel tempo seguano una distribuzione binomiale e l'obiettivo del test di Kupiec è quello di determinare la consistenza di queste violazioni con un dato livello di confidenza. Se il numero di superamenti osservati differisce di molto da quelli attesi, allora l'adeguatezza del modello deve essere verificata. Per effettuare il test, abbiamo bisogno del numero effettivo di violazioni (E), numero di osservazioni utilizzate per il forecast (N) e la probabilità del VaR (p). Assumendo, quindi, che E sia distribuito come una Binomiale (N, p), l'ipotesi nulla è:

$$H_0: p = p_0, \quad p_0 = 0{,}01 \; o \; 0{,}05$$

e p è stimato come $\frac{E}{N}$. Quindi, andiamo a testare se il numero di violazioni osservate sia significativamente differente da quello atteso. Kupiec (1995) propose questo test[20]:

$$-2\ln[(1-p)^{N-E}p^{E}] + 2\ln[(1-p_0)^{N-E}{p_0}^{E}]$$

che segue una distribuzione chi-quadro con un grado di libertà.

3.6.3 Test di Christoffersen

Se le variazioni giornaliere del portafoglio fossero indipendenti, le violazioni dovrebbero essere distribuite uniformemente per tutto il periodo utilizzato per i test retrospettivi (backtesting). In pratica, sono spesso raggruppati suggerendo che le perdite nei giorni successivi non sono indipendenti (John C. Hull, 2015)[21]. Un modo per verificare ciò è il test statistico proposto da Christoffersen (1998):

[20] P. Kupiec, "Techniques for Verifying the Accuracy of Risk Management Models," Journal of Derivatives 3 (1995): 73–84.

[21] John C. Hull, Risk management and financial institutions (quarta edizione), Wiley (2015).

$$-2\ln[(1-\pi)^{u_{00}+u_{10}}\pi^{u_{01}+u_{11}}]+2\ln\left[(1-\pi_{01})^{u_{00}}\pi_{01}^{u_{01}}(1-\pi_{11})^{u_{10}}\pi_{11}^{u_{11}}\right]$$

dove u_{ij} è il numero di osservazioni in cui si va da un giorno in cui siamo nello stato i ad un giorno in cui siamo nello stato j. Lo stato 0 è il giorno in cui non ci sono violazioni mentre lo stato 1 è il giorno dove si verifica una violazione. Inoltre, π, π_{01}, π_{11} sono definiti come (John C. Hull, 2015):

$$\pi = \frac{u_{01}+u_{11}}{u_{00}+u_{01}+u_{10}+u_{11}}$$

$$\pi_{01} = \frac{u_{01}}{u_{00}+u_{01}}$$

$$\pi_{11} = \frac{u_{11}}{u_{10}+u_{11}}$$

3.7 Backtesting

In questo paragrafo andremo a svolgere la procedura di backtesting per verificare la bontà delle previsioni. Per fare ciò definiamo le nostre *estimation window* e *testing window*. La serie dei rendimenti del portafoglio è composta dal 1578 osservazioni che vanno dal 07.08.2015 al 01.12.2019. Prendiamo le prime 1078 osservazioni che chiamiamo *estimation window* e manteniamo le ultime 500 osservazioni fuori dal campione (*testing window*). La nostra *estimation window* consiste in osservazione da 1 a 1078 compreso e utilizzando questo campione stimiamo i valori del VaR, con un livello di confidenza del 95% e del 99%, per il giorno successivo (1079-esimo). In seguito, la nostra *estimation window* viene spostata di un giorno permettendoci di fare previsioni sul VaR per il 1080-esimo giorno. Pertanto, ripetendo ciò per 500 volte (la grandezza della nostra *testing window*) otteniamo le nostre stime del VaR per un periodo di 500 giorni. Utilizzando lo stesso procedimento per tutti i modelli GARCH utilizzati e per le *historical simulation*, siamo in grado di verificare la bontà delle previsioni con i tre metodi spiegati nei paragrafi precedenti (violation ratio, test di Kupiec e test di Christoffersen). In tabella 13 e 14 guardiamo ai risultati di tali test nel caso, rispettivamente, di un livello di confidenza del 99% e del 95%. Nel primo caso ($X = 99\%$), il numero atteso di casi di superamento è 5 ($p * N = 0{,}01 * 500$). Il rapporto di violazione (vd. (51)) è sempre lo stesso (0,8) per tutti i modelli GARCH stimati. Invece, nel caso di *historical simulation* il rapporto di violazione è 1,2. Da una prima analisi, quindi, potremmo dire che i modelli sono dei buoni stimatori del value at risk, in quanto, secondo la prassi, il valore ottimale del rapporto di violazione deve trovarsi tra 0,8 e 1,2. Tuttavia, per trovare maggiore conferma a quanto affermato utilizziamo i test di Kupiec e di Christoffersen. Entrambi i test riportano dei p-values molto elevati.

Pertanto, non è possibile rifiutare l'ipotesi nulla che prevede un corretto numero di violazioni. Nel test di Christoffersen (come già spiegato nel par. 3.6.3) va a testare anche l'indipendenza delle violazioni. Quindi, non potendo rifiutare l'ipotesi nulla del test di Christoffersen, si sta sì affermando che il numero delle violazioni è corretto ma il test ci dice anche che tali violazioni sono indipendenti tra di loro. Nel secondo caso ($X = 95\%$), il numero atteso di casi di superamento è 25 ($p * N = 0{,}05 * 500$). Come è possibile vedere in tabella 14, i risultati ottenuti sono diversi tra i vari modelli GARCH stimati. Tutti i modelli GARCH sembrano essere dei buoni stimatori del value at risk, in quanto i valori del rapporto di violazione si trova tra 0,8 e 1,2 e soprattutto perché, anche in questo caso, non è possibile rifiutare l'ipotesi nulla dei test statistici di Kupiec e Christoffersen. Tuttavia, il miglior modello sembra essere l'SGARCH che ha un rapporto di violazione uguale a 1. Invece, per quanto riguarda il caso di *historical simulation*, il rapporto di violazione è di 0,48. Effettuando i test statistici otteniamo dei p-values molto bassi per i quali siamo costretti a rifiutare l'ipotesi nulla.

	VaR (95%)	**VaR (99%)**
EGARCH	-0,0586	-0,121
GJR-GARCH	-0,0521	-0,108
TGARCH	-0,0587	-0,122
SGARCH	-0,0521	-0,108
IGARCH	-0,0522	-0,108

Tabella 12: VaR portafoglio calcolato con i modelli GARCH con X = 95% e X = 99%

	99%		
	VR	**Test di Kupiec**	**Test di Christoffersen**
EGARCH	0,8	Non rifiuto H_0	Non rifiuto H_0
GJR-GARCH	0,8	Non rifiuto H_0	Non rifiuto H_0
TGARCH	0,8	Non rifiuto H_0	Non rifiuto H_0
SGARCH	0,8	Non rifiuto H_0	Non rifiuto H_0
IGARCH	0,8	Non rifiuto H_0	Non rifiuto H_0
Historical simulation	1,2	Non rifiuto H_0	Non rifiuto H_0

Tabella 13: Violation ratio, test di Kupiec e test di Christoffersen con X = 99%

95%			
	VR	**Test di Kupiec**	**Test di Christoffersen**
EGARCH	0,92	Non rifiuto H_0	Non rifiuto H_0
GJR-GARCH	1,08	Non rifiuto H_0	Non rifiuto H_0
TGARCH	0,92	Non rifiuto H_0	Non rifiuto H_0
SGARCH	1	Non rifiuto H_0	Non rifiuto H_0
IGARCH	0,96	Non rifiuto H_0	Non rifiuto H_0
Historical simulation	0,48	Rifiuto H_0	Rifiuto H_0

Tabella 14: Violation ratio, test di Kupiec e test di Christoffersen con X = 95%

Conclusione

In questa tesi abbiamo visto come sia diventato importante analizzare il momento secondo di una variabile. Soprattutto dopo la crisi finanziaria del 2007-2009, le regole riguardo la valutazione dei rischi sono diventate sempre più stringenti per Banche e grandi istituti finanziari che hanno dovuto implementare modelli sempre più accurati. Quindi, si può ben capire come la modellizzazione e la previsione della volatilità o, in altre parole, la struttura della covarianza dei rendimenti degli asset sia di primaria importanza per tali soggetti.

Quindi, tali analisi portano ad un miglioramento delle decisioni prese in varie aree, come ad esempio in *asset pricing, portfolio optimization, option pricing, hedging and risk management*. In questo elaborato, abbiamo preso come variabili in esame le criptovalute. A differenza delle valute tradizionali, le criptovalute si basano sulla prova crittografica, che fornisce molti vantaggi rispetto ai metodi di pagamento tradizionali (come le carte di credito) tra cui alta liquidità, minori costi di transazione, e l'anonimato. L'interesse per Bitcoin e altre criptovalute è aumentato notevolmente negli ultimi anni. L'analisi svolta può essere divisa in quattro parti. Nella prima parte, svolgiamo un'analisi del Bitcoin modellizzando la sua volatilità condizionata attraverso diversi GARCH univariati e con tre diverse distribuzioni. Come distribuzioni utilizziamo la t di student, la GED, e la normale inversa gaussiana per catturare l'eccesso di curtosi e di skewness presente nella serie dei rendimenti. Il modello scelto è un TGARCH con distribuzione normale inversa gaussiana nonostante non sia quello con Akaike, BIC, Shibata e Hannan-Quinn più basso. Questo perché è l'unico modello, insieme al GARCH integrato con distribuzione GED, che non presenta autocorrelazioni seriali nei residui standardizzati e nei residui al quadrato. Infine, il TGARCH è preferito rispetto all'IGARCH in quanto presenta più parametri

statisticamente significativi. La seconda parte dell'analisi si svolge in campo multivariato, analizzando quattro criptovalute: Bitcoin, Ethereum, Litecoin e Monero. Scegliamo un modello DCC-GARCH per la semplicità del processo di stima diviso in due fasi. La prima fase di stima di GARCH univariati e la seconda fase consiste invece nella stima della matrice di correlazione R_t. La stima del modello è in figura 21. I modelli GARCH univariati, utilizzati per la prima fase del processo di stima, sono gli stessi usati nel caso dell'analisi del Bitcoin, ovvero SGARCH, TGARCH, GJR-GARCH, IGARCH e EGARCH. Però in questo caso viene utilizzata solo la distribuzione t di student. I valori di Akaike e degli altri criteri sono molto simili tra di loro, ma il modello EGARCH è quello che presenta più parametri statisticamente significativi. Inoltre, il modello EGARCH riesce a catturare anche l'effetto leverage che è presente nelle serie. I parametri del DCC sono anch'essi statisticamente significativi e quindi l'EGARCH viene scelto come modello migliore. Nell'analisi della correlazione, vediamo che le varie criptovalute prese a due a due sono molto correlate tra di loro. Però questa correlazione è poco accentuata nel periodo 2015-2017 (assumendo addirittura valori negativi anche se solo per pochi giorni), ma da metà-fine anno 2017 la correlazione cresce esponenzialmente fino a toccare valori vicino all'unità. Questo è dovuto forse al fatto che nel 2017 è cresciuto molto l'interesse degli investitori nei confronti del Bitcoin (c'è stato il boom del prezzo del Bitcoin che nel corso del 2017 ha raggiunto valori intorno ai 20000 $) e delle criptovalute in generale e ciò ha fatto crescere la correlazione fra le varie criptovalute. Nella terza parte dell'analisi, grazie alle stime dei rendimenti attesi e delle covarianze e varianze attese ottenute con i forecast del modello DCC-GARCH, costruiamo il nostro portafoglio di criptovalute composto naturalmente da Bitcoin, Ethereum, Litecoin e Monero. Con l'aiuto del pacchetto parma (portfolio allocation and risk management application) su R, riusciamo a calcolare i pesi ottimali del nostro portafoglio, rappresentando la nostra frontiera efficiente (figura 30). Notiamo, inoltre, che se non imponiamo vincoli ai pesi nel processo di ottimizzazione, il portafoglio che si ottiene (portafoglio ottimale non vincolato) è preferibile a quello vincolato. Tuttavia, senza l'utilizzo di vincoli è possibile che vengano prodotti dei portafogli troppo concentrati e questo è molto rischioso. Infatti, dato che non c'è certezza che il modello e le stime siano esatte, la diversificazione è una contromisura per difendersi da eventi non prevedibili. Infine, nella quarta parte dell'analisi, andiamo a stimare il value at risk della serie di rendimenti del portafoglio creato con i pesi ottimali ottenuti in precedenza. Per la stima del VaR si ipotizza una media nulla, e viene definito come $\mathrm{VaR} = \sigma_{t+1} * \mathrm{F}^{-1}(\alpha)$ dove σ_{t+1} è la varianza stimata al tempo $t+1$ attraverso un forecast, e $\mathrm{F}^{-1}(\alpha)$ è il valore critico della distribuzione t di student. I VaR sono calcolati con un livello di confidenza del 95% e del 99%. Le metodologie utilizzate per la stima sono due: *historical simulation* e l'utilizzo di GARCH univariati che permettono appunto la stima della varianza condizionata al tempo $t+1$. Il primo metodo usa osservazioni passate dei rendimenti per effettuare previsioni del VaR. Esso consiste in tre steps:

- ordinare la serie dei rendimenti del portafoglio dal valore più piccolo a quello più grande
- va trovato il p% quantile della serie ordinata. Ad esempio, se $p = 0{,}01$ (in caso di VaR (99%)) allora $q = p * N$ dove N è il numero delle osservazioni. Nel nostro caso, $q = 0{,}01 * 1577 = 15{,}77 \sim 16$.
- estraiamo il q-esimo valore dalla serie ordinata, in questo caso il 16-esimo.

Inoltre, per verificare la bontà delle previsioni del VaR effettuate dalle due metodologie, utilizziamo dei diversi test statistici: violation ratio, test di Kupiec e test di Christoffersen. Da questi test ricaviamo che i GARCH univariati sono in grado di fare delle stime più accurate del VaR rispetto al metodo *historical simulation*. In particolare, vediamo che, nel caso di un livello di confidenza del 99%, le stime dei diversi GARCH univariati hanno la stessa bontà previsionale. Inoltre, in questo caso anche il metodo HS è un buon stimatore. Invece, nel caso di un livello di confidenza del 95%, i modelli GARCH sono tutti dei buoni stimatori, però il modello SGARCH è quello che presenta un rapporto di violazione di 1 quindi sembra essere il miglior modello. Invece, per quanto riguarda il metodo HS, dai test statistici ricaviamo che non è un buon modello previsionale.
Ulteriori studi in tal senso potrebbero essere svolti affiancando all'analisi delle sole criptovalute anche altre serie finanziarie come gli indici azionari, analizzando le correlazioni tra le criptovalute e tali asset al fine di costruire un portafoglio di investimento più diversificato.

Bibliografia:

- L. Bauwens et al., Multivariate GARCH models: A survey, Journal of Applied Econometrics,21: 79-106 (2006).
- T. Bollerslev, Generalized Autoregressive Conditional Heteroskedasticity, Journal of Econometrics, 31: 307-327 (1986).
- Daniel B. Nelson, Conditional Heteroskedasticity in Asset returns: A new approach, Econometrica, vol. 59: 347-370 (1991).
- Daniel B. Nelson et al., Inequality Constraints in the Univariate GARCH Model, Journal of Business & Economic Statistics, vol. 10: 229-235 (1992).
- T. Bollerslev et al., A Capital Asset Pricing Model with Time-Varying Covariances, Journal of Political Economy, vol. 96: 116-131 (1988).
- Franses e van Dijk, Non-Linear Time Series Models in Empirical Finance, Cambridge Books from Cambridge University Press (2000).
- Rober F. Engle et al., Multivariate simultaneous Generalized ARCH, Econometric Theory, 11: 122-150 (1995).
- Rober F. Engle et al., Asset Pricing with a Factor-ARCH covariance structure: Empirical Estimates for Treasury Bills, Journal of Econometrics, 45: 213-237 (1990).
- Zouheir Mighri et al., Volatility Spillovers among the Cryptocurrency Time Series, International Journal of Economics and Financial, 9 (3): 81-90 (2019).
- Danielsson, J., Financial Risk Forecasting. WILEY, Londra (2011).
- Dowd, K., Measuring market risk. WILEY, Londra (2002).
- John C. Hull, Risk management and financial institutions (quarta edizione), Wiley (2015).
- Richmond O. Siaw, Investment Portfolio Optimization with GARCH Models, Elk Asia Pacific Journal of Finance and Risk Management, Volume 8 (2017).
- Robert F. Engle, Dynamic Conditional Correlation: A Simple Class of Multivariate GARCH Models, Forthcoming Journal of Business and Economic Statistics (2002).
- Chu, J.; Chan, S.; Nadarajah, S.; Osterrieder, J. GARCH Modelling of Cryptocurrencies. J. Risk Financial Management, 2017, 10, 17

Appendice A.1: Codice R studio - Analisi univariata Bitcoin

Analasi univariata Bitcoin

 # Installazione pacchetti R

```
install.packages("quantmod"); install.packages("rmgarch"); install.packages("tseries");
install.packages("fBasics"); install.packages("graphics"); install.packages("MTS");

library("quantmod"); library("rmgarch"); library("vars"); library("tseries"); library("fBasics");
library("graphics"); library("MTS").
```

 # Price data

```
startDate= as.Date("2015-08-07")
endDate= as.Date("2019-12-01")
getSymbols ("BTC-USD", from= startDate, to= endDate)
```

 # Plot

```
chartSeries(`BTC-USD`)
```

 # BTC price

```
btc_price = `BTC-USD` [,4]
colnames (btc_price) = c('btc_price')
basicStats (btc_price)
```

 # ACF - PACF Bitcoin

```
par (mfrow=c (2,1))
acf((btc_price), lag.max = NULL, main = "bitcoin")
pacf((btc_price), lag.max = NULL, main= "bitcoin")
dev.off ()
```

#Returns

```
btc_ret= dailyReturn(`BTC-USD`)
colnames(btc_ret) = c('btc_ret')
plot (btc_ret, type= 'l')
```

 # Analisi univariata bitcoin

```
basicStats(btc_ret)
par (mfrow= c (2,1))
acf((btc_ret), lag.max = NULL, main = "bitcoin")
pacf((btc_ret), lag.max = NULL, main= "bitcoin")
dev.off ()
```

 # Test per la stazionarietà

```
adf.test(btc_ret)$p.value
adf.test(btc_ret)
PP.test(btc_ret) # Serie stazionaria
PP.test(btc_ret)$p.value
```

#Arima (3, 0, 3) miglior modello

```
arima303= arima (btc_ret, order= c (3,0,3)); arima303
coeftest(arima303)
show(arima303)
```

 # test sulle autocorrelazioni dei residui al quadrato- ARCH EFFECTS

```
res_arima303= arima303$residuals
residuals_btc_square= res_arima303^2
Box.test(residuals_btc_square, lag=20, type="Ljung-Box")

par (mfrow= c (2,1))
acf((residuals_btc_square), lag.max = NULL, main = "residuals_btc_square")
pacf((residuals_btc_square), lag.max = NULL, main= "residuals_btc_square")
dev.off()
basicStats(residuals_btc_square)
```

#Univariato Garch bitcoin

 # sGarch (1,1) t-stud AIC= -10.626, BIC= -10.609, shibata= -10.626, Hannan-Quinn=-10.620

```
fit.Stdspec= ugarchspec(variance.model = list(model = "sGARCH", garchOrder = c(1,1),
submodel = NULL, external.regressors= NULL, variance.targeting = FALSE), mean.model = list(armaOrder
= c(0,0), include.mean = TRUE, archm = FALSE, archpow = 1, arfima = FALSE, external.regressors =
NULL, archex = FALSE),distribution.model = "std", start.pars = list(), fixed.pars = list())
fit.GARCHStd = ugarchfit(data = residuals_btc_square, spec = fit.Stdspec, solver="hybrid")
```

```
show(fit.GARCHStd)
plot(fit.GARCHStd)
```

 # sGarch (1,1) GED AIC= -11.459, BIC= -11.442, shibata= -11.459, quinn=-11.453

```
fit.Gedspec= ugarchspec(variance.model = list(model = "sGARCH", garchOrder = c(1,1), submodel =
NULL, external.regressors= NULL, variance.targeting = FALSE), mean.model = list(armaOrder = c(0,0),
include.mean = TRUE, archm = FALSE, archpow = 1, arfima = FALSE, external.regressors = NULL,
archex = FALSE), distribution.model = "ged", start.pars = list(), fixed.pars = list())
fit.GARCHGed = ugarchfit(data = residuals_btc_square, spec = fit.Gedspec, solver="hybrid")
show(fit.GARCHGed)
```

 # sGarch (1,1) NIG AIC= -11.497, BIC= -11.477, shibata=-11.497, quinn=-11.489

```
fit.Nigspec= ugarchspec(variance.model = list(model = "sGARCH", garchOrder = c(1,1), submodel =
NULL, external.regressors= NULL, variance.targeting = FALSE), mean.model = list(armaOrder = c(0,0),
include.mean = TRUE, archm = FALSE, archpow = 1, arfima = FALSE, external.regressors = NULL,
archex = FALSE), distribution.model = "nig", start.pars = list(), fixed.pars = list())
fit.GARCHNig = ugarchfit(data = residuals_btc_square, spec = fit.Nigspec, solver="hybrid")
show(fit.GARCHNig)
```

 # eGARCH(1,1) t-stud AIC= -10.668, BIC= -10.648, shibata=-10.668,quinn=-10.660

```
fit.Stdespec <- ugarchspec(variance.model = list(model = "eGARCH", garchOrder = c(1, 1), submodel =
NULL, external.regressors = NULL, variance.targeting = FALSE), mean.model = list(armaOrder = c(0, 0),
include.mean = TRUE, archm = FALSE, archpow = 1, arfima = FALSE, external.regressors = NULL,
archex = FALSE), distribution.model = "std", start.pars = list(), fixed.pars = list())
fiteGARCHStd = ugarchfit(data = residuals_btc_square, spec = fit.Stdespec, solver= "hybrid")
show(fiteGARCHStd)
plot(fiteGARCHStd)
StdegarchSigma= fiteGARCHStd@fit$sigma
```

 # eGARCH(1,1) GED AIC= -11.463, BIC= -11.443, shibata=-11.463,quinn=-11.456

```
fit.GEDespec <- ugarchspec(variance.model = list(model = "eGARCH", garchOrder = c(1, 1), submodel =
NULL, external.regressors = NULL, variance.targeting = FALSE), mean.model = list(armaOrder = c(0, 0),
include.mean = TRUE, archm = FALSE,  archpow = 1, arfima = FALSE, external.regressors = NULL,
archex = FALSE), distribution.model = "ged", start.pars = list(), fixed.pars = list())
fiteGARCHGED = ugarchfit(data = residuals_btc_square, spec = fit.GEDespec,  solver= "hybrid")
show(fiteGARCHGED)
```

 # eGARCH(1,1) NIG AIC= -11.864, BIC= -11.840, shibata=-11.864,quinn=-11.855

```
fit.Nigespec <- ugarchspec(variance.model = list(model = "eGARCH", garchOrder = c(1, 1), submodel =
NULL, external.regressors = NULL, variance.targeting = FALSE), mean.model = list(armaOrder = c(0, 0),
include.mean = TRUE, archm = FALSE, archpow = 1, arfima = FALSE, external.regressors = NULL,
archex = FALSE), distribution.model = "nig", start.pars = list(), fixed.pars = list())
fiteGARCHNig = ugarchfit(data = residuals_btc_square, spec = fit.Nigespec,  solver= "hybrid" )
show(fiteGARCHNig)
plot(fiteGARCHNig)
```

#IGARCH (1,1) std AIC= -10.732, BIC= -10.718, shibata=-10.732, quinn=-10.727

```
fit.stdispec <- ugarchspec(variance.model = list(model = "iGARCH", garchOrder = c(1, 1), submodel =
NULL, external.regressors = NULL, variance.targeting = FALSE), mean.model = list(armaOrder = c(0, 0),
include.mean = TRUE, archm = FALSE, archpow = 1, arfima = FALSE, external.regressors = NULL,
archex = FALSE), distribution.model = "std", start.pars = list(), fixed.pars = list())
fitIGARCHstd = ugarchfit(data = residuals_btc_square, spec = fit.stdispec, solver ="hybrid")
show(fitIGARCHstd)
plot(fitIGARCHstd)
iResStd = residuals(fitIGARCHstd)
stdigarchSigma=fitIGARCHstd@fit$sigma
```

#IGARCH (1,1) ged AIC= -11.444, BIC= -11.430, shibata=-11.444, quinn=-11.439

```
fit.gedispec <- ugarchspec(variance.model = list(model = "iGARCH", garchOrder = c(1, 1), submodel =
NULL, external.regressors = NULL, variance.targeting = FALSE), mean.model = list(armaOrder = c(0, 0),
include.mean = TRUE, archm = FALSE, archpow = 1, arfima = FALSE, external.regressors = NULL,
archex = FALSE), distribution.model = "ged", start.pars = list(), fixed.pars = list())
fitIGARCHged = ugarchfit(data = residuals_btc_square, spec = fit.gedispec, solver ="hybrid")
show(fitIGARCHged)
plot(fitIGARCHged)
iResGed = residuals(fitIGARCHged)
gedigarchSigma=fitIGARCHged@fit$sigma
```

#IGARCH (1,1) nig AIC= -11.191, BIC= -11.174, shibata=-11.191, quinn=-11.185

```
fit.nigispec <- ugarchspec(variance.model = list(model = "iGARCH", garchOrder = c(1, 1), submodel =
NULL, external.regressors = NULL, variance.targeting = FALSE), mean.model = list(armaOrder = c(0, 0),
```

include.mean = TRUE, archm = FALSE, archpow = 1, arfima = FALSE, external.regressors = NULL, archex = FALSE), distribution.model = "nig", start.pars = list(), fixed.pars = list())
fitIGARCHnig = ugarchfit(data = residuals_btc_square, spec = fit.nigispec, solver ="hybrid")
show(fitIGARCHnig)
plot(fitIGARCHnig)
iResNig = residuals(fitIGARCHnig)
nigigarchSigma=fitIGARCHnig@fit$sigma

 # GJRGARCH std AIC=-10.735, BIC=-10.715, shibata=-10.735, quinn=-10.727
fit.stdGJRspec <- ugarchspec(variance.model = list(model = "gjrGARCH", garchOrder = c(1, 1), submodel = NULL, external.regressors = NULL, variance.targeting = FALSE), mean.model = list(armaOrder = c(0, 0), include.mean = TRUE, archm = FALSE, archpow = 1, arfima = FALSE, external.regressors = NULL, archex = FALSE), distribution.model = "std", start.pars = list(), fixed.pars = list())
fitGJRGARCHstd = ugarchfit(data = residuals_btc_square, spec = fit.stdGJRspec, solver="hybrid")
show(fitGJRGARCHstd)
plot(fitGJRGARCHstd)
stdGJRres = residuals(fitGJRGARCHstd)
GJRgarchSigmaStd=fitGJRGARCHstd@fit$sigma

 # GJRGARCH GED N/A
fit.GEDGJRspec <- ugarchspec(variance.model = list(model = "gjrGARCH", garchOrder = c(1, 1), submodel = NULL, external.regressors = NULL, variance.targeting = FALSE), mean.model = list(armaOrder = c(0, 0), include.mean = TRUE, archm = FALSE, archpow = 1, arfima = FALSE, external.regressors = NULL, archex = FALSE), distribution.model = "ged", start.pars = list(), fixed.pars = list())
fitGJRGARCHGED = ugarchfit(data = residuals_btc_square, spec = fit.GEDGJRspec, fit.control=list(fixed.se=1))
show(fitGJRGARCHGED)
plot(fitGJRGARCHGED)
GEDGJRres = residuals(fitGJRGARCHGED)
GJRgarchSigmaGED=fitGJRGARCHGED@fit$sigma

 # GJRGARCH nig AIC=-11.746, BIC=-11.722, shibata=-11.746, quinn=-11.737
fit.nigGJRspec <- ugarchspec(variance.model = list(model = "gjrGARCH", garchOrder = c(1, 1), submodel = NULL, external.regressors = NULL, variance.targeting = FALSE), mean.model = list(armaOrder = c(0, 0),

include.mean = TRUE, archm = FALSE, archpow = 1, arfima = FALSE, external.regressors = NULL, archex = FALSE), distribution.model = "nig", start.pars = list(), fixed.pars = list())
fitGJRGARCHnig = ugarchfit(data = residuals_btc_square, spec = fit.nigGJRspec, solver="hybrid")
show(fitGJRGARCHnig)
plot(fitGJRGARCHnig)
NigGJRres = residuals(fitGJRGARCHnig)
GJRgarchSigmaNig=fitGJRGARCHnig@fit$sigma

 # TGARCH t-stud AIC= -10.798, BIC= -10.778, shibata=-10.798, quinn=-10.791
fit.stdTspec <- ugarchspec(variance.model = list(model = "fGARCH", garchOrder = c(1, 1), submodel = "TGARCH", external.regressors = NULL, variance.targeting = FALSE), mean.model = list(armaOrder = c(0, 0), include.mean = TRUE, archm = FALSE, archpow = 1, arfima = FALSE, external.regressors = NULL, archex = FALSE), distribution.model = "std", start.pars = list(), fixed.pars = list())
fitTGARCHstd = ugarchfit(data = residuals_btc_square, spec = fit.stdTspec, solver="hybrid")
show(fitTGARCHstd)
plot(fitTGARCHstd)
stdTres = residuals(fitTGARCHstd)
TgarchSigmaStd=fitTGARCHstd@fit$sigma

 # TGARCH GED AIC= -1.6103, BIC= -1.5899, shibata=-1.6103, quinn=-1.6027
fit.GEDTspec <- ugarchspec(variance.model = list(model = "fGARCH", garchOrder = c(1, 1), submodel = "TGARCH", external.regressors = NULL, variance.targeting = FALSE), mean.model = list(armaOrder = c(0, 0), include.mean = TRUE, archm = FALSE, archpow = 1, arfima = FALSE, external.regressors = NULL, archex = FALSE), distribution.model = "nig", start.pars = list(), fixed.pars = list())
fitTGARCHGED = ugarchfit(data = residuals_btc_square, spec = fit.GEDTspec, solver="hybrid")
show(fitTGARCHGED)
plot(fitTGARCHGED)
TGEDres = residuals(fitTGARCHGED)
TgarchSigmaGED=fitTGARCHGED@fit$sigma

 # TGARCH nig AIC= -11.278, BIC= -11.255, shibata=-11.278, quinn=-11.269
fit.NIGTspec <- ugarchspec(variance.model = list(model = "fGARCH", garchOrder = c(1, 1), submodel = "TGARCH", external.regressors = NULL, variance.targeting = FALSE), mean.model = list(armaOrder = c(0, 0), include.mean = TRUE, archm = FALSE, archpow = 1, arfima = FALSE, external.regressors = NULL, archex = FALSE), distribution.model = "nig", start.pars = list(), fixed.pars = list())
fitTGARCHNIG = ugarchfit(data = residuals_btc_square, spec = fit.NIGTspec, solver="hybrid")

```
show(fitTGARCHNIG)
plot(fitTGARCHNIG)
TNIGres = residuals(fitTGARCHNIG)
TgarchSigmaNIG=fitTGARCHNIG@fit$sigma
```

Appendice A.2: Codice R studio – Analisi multivariata – Modello DCC

```
## Multivariato ##
Installazione pacchetti R
install.packages("quantmod"); install.packages("rmgarch"); install.packages("tseries");
install.packages("fBasics"); install.packages("graphics"); install.packages("MTS");
install.packages("lmtest");

library("quantmod"); library("rmgarch"); library("vars"); library("tseries"); library("fBasics");
library("graphics"); library("MTS"); library("lmtest").

# Price data
startDate= as.Date("2015-08-07")
endDate= as.Date("2019-12-01")

getSymbols ("BTC-USD", from= startDate, to= endDate)
getSymbols ("ETH-USD", from= startDate, to= endDate)
getSymbols ("LTC-USD", from= startDate, to= endDate)
getSymbols("XMR-USD", from= startDate, to= endDate)
# Plot
chartSeries(`BTC-USD`)
chartSeries(`ETH-USD`)
chartSeries(`LTC-USD`)
chartSeries(`XMR-USD`)
# BTC price
btc_price= `BTC-USD` [,4]
colnames(btc_price) = c('btc_price')
basicStats(btc_price)

# ACF - PACF Bitcoin
```

```
par (mfrow= c (2,1))
acf((btc_price), lag.max = NULL, main = "bitcoin")
pacf((btc_price), lag.max = NULL, main= "bitcoin")
dev.off ()
```

#ETH price

```
eth_price= `ETH-USD` [,4]
colnames(eth_price) = c('eth_price')
basicStats(eth_price)
```

#ACF - PACF Ethereum

```
par (mfrow= c (2,1))
acf((eth_price), lag.max = NULL, main = "Ethereum")
pacf((eth_price), lag.max = NULL, main= "Ethereum")
dev.off ()
```

 # Litecoin price

```
ltc_price= `LTC-USD` [,4]
colnames(ltc_price) = c('ltc_price')
basicStats(ltc_price)
```

 # ACF - PACF Litecoin

```
par(mfrow=c(2,1))
acf((ltc_price), lag.max = NULL, main = "Litecoin")
pacf((ltc_price), lag.max = NULL, main= "Litecoin")
dev.off()
```

 # Monero price

```
mon_price= `XMR-USD`[,4]
colnames(mon_price)= c('mon_price')
basicStats(mon_price)
```

 # ACF - PACF Monero

```
par(mfrow=c(2,1))

acf((mon_price), lag.max = NULL, main = "Monero")
```

```
pacf((mon_price), lag.max = NULL, main= "Monero")

dev.off()

#Returns
btc_ret= dailyReturn(`BTC-USD`)
eth_ret= dailyReturn(`ETH-USD`)
ltc_ret= dailyReturn(`LTC-USD`)
mon_ret= dailyReturn(`XMR-USD`)

colnames(btc_ret)=c('btc_ret')
basicStats(btc_ret)
colnames(eth_ret)=c('eth_ret')
basicStats(eth_ret)
colnames(ltc_ret)=c('ltc_ret')
basicStats(ltc_ret)
colnames(mon_ret)= c('mon_ret')
basicStats(mon_ret)

plot(btc_ret, type= 'l')
plot(eth_ret, type= 'l')
plot(ltc_ret, type= 'l')
plot(mon_ret, type= 'l')

# Test per la stazionarietà
adf.test(btc_ret)$p.value
adf.test(btc_ret)
PP.test(btc_ret) # Serie stazionaria
PP.test(btc_ret)$p.value

adf.test(eth_ret)$p.value
adf.test(eth_ret)
PP.test(eth_ret) # Serie stazionaria

adf.test(ltc_ret)$p.value
```

```
adf.test(ltc_ret)
PP.test(ltc_ret) # Serie stazionaria

adf.test(mon_ret)$p.value
adf.test(mon_ret)
PP.test(mon_ret) # Serie stazionaria

# Multivariato
# Matrice dei returns= rX
rX= data.frame(btc_ret= btc_ret, eth_ret= eth_ret, ltc_ret= ltc_ret, mon_ret= mon_ret)
View(rX)
# Modelliamo la media con un VAR model
mean_var= VARMACpp(rX, p = 1, q = 0, include.mean = T, fixed = NULL)
summary(mean_var)
#granger test
grangertest(btc_ret, ltc_ret, order=5)
grangertest(btc_ret, eth_ret, order=5)
grangertest(ltc_ret, eth_ret, order=5)
grangertest(btc_ret, mon_ret, order=5)
grangertest(ltc_ret, mon_ret, order=5)
grangertest(eth_ret, mon_ret, order=5)
#estraggo media condizionata e residui
res=mean_var[["residuals"]]
coef= mean_var[["Phi"]]
ret_X= rX[1578,]
ret_x= t(ret_X)
cond.mean= (coef%*%ret_x)

acf((res), lag.max = NULL, main = "residuals")
pacf((res), lag.max = NULL, main = "residuals")

sres= res^2
values = seq(from = as.Date("2015-08-08"), to = as.Date("2019-12-01"), by = 'day')
sres= data.frame(sres, row.names= values)

acf((sres), lag.max = NULL, main = "sresiduals")
```

```
pacf((sres), lag.max = NULL, main = "sresiduals")
```

#DCC, stima a due passaggi

#DCC igarch mvt AIC= -36.217, BIC= -36.132, Shibata=-36.218, Quinn=-36.186

```
uspec.i= multispec(replicate(4, ugarchspec(variance.model = list(model = "iGARCH", garchOrder = c(1, 1),
submodel = NULL, external.regressors = NULL, variance.targeting = FALSE), mean.model =
list(armaOrder = c(0, 0), include.mean = TRUE, archm = FALSE, archpow = 1, arfima = FALSE,
external.regressors = NULL, archex = FALSE), distribution.model = "std", start.pars = list(), fixed.pars =
list())))

multf.i= multifit(uspec.i, sres)
show(multf.i)
spec.i= dccspec(uspec= uspec.i, dccOrder = c(1,1), distribution = 'mvt')
fit.i= dccfit(spec.i, data= sres,  fit= multf.i)
show(fit.i)
```

#DCC GJR-garch mvt AIC= -36.194, BIC= -36.081, Shibata=-36.195, Quinn=-36.152

```
uspec.gjr= multispec(replicate(4, ugarchspec(variance.model = list(model = "gjrGARCH", garchOrder = c(1,
1), submodel = NULL, external.regressors = NULL, variance.targeting = FALSE), mean.model =
list(armaOrder = c(0, 0), include.mean = TRUE, archm = FALSE, archpow = 1, arfima = FALSE,
external.regressors = NULL, archex = FALSE), distribution.model = "std", start.pars = list(), fixed.pars =
list())))
multf.gjr= multifit(uspec.gjr, sres)
show(multf.gjr)
spec.gjr= dccspec(uspec= uspec.gjr, dccOrder = c(1,1), distribution = 'mvt')
fit.gjr= dccfit(spec.gjr, data= sres,  fit= multf.gjr)
show(fit.gjr)
```

#DCC egarch mvt AIC= -36.092, BIC= -35.980, Shibata=-36.093, Quinn=-36.050

```
uspec.e= multispec(replicate(4, ugarchspec(variance.model = list(model = "eGARCH", garchOrder = c(1, 1),
```

```
submodel = NULL, external.regressors = NULL, variance.targeting = FALSE), mean.model =
list(armaOrder = c(0, 0), include.mean = TRUE, archm = FALSE, archpow = 1, arfima = FALSE,
external.regressors = NULL, archex = FALSE), distribution.model = "std", start.pars = list(), fixed.pars =
list())))
multf.e= multifit(uspec.e, sres)
show(multf.e)
spec.e= dccspec(uspec= uspec.e, dccOrder = c(1,1), distribution = 'mvt')
fit.e= dccfit(spec.e, data= sres,  fit= multf.e)
show(fit.e)
```

#DCC sgarch mvt AIC= -36.269, BIC= -36.170, Shibata=-36.269, Quinn=-36.232

```
uspec.s= multispec(replicate(4, ugarchspec(variance.model = list(model = "sGARCH", garchOrder = c(1, 1),
submodel = NULL, external.regressors = NULL, variance.targeting = FALSE), mean.model =
list(armaOrder = c(0, 0), include.mean = TRUE, archm = FALSE,  archpow = 1, arfima = FALSE,
external.regressors = NULL, archex = FALSE), distribution.model = "std", start.pars = list(), fixed.pars =
list())))
multf.s= multifit(uspec.s, sres)
show(multf.s)
spec.s= dccspec(uspec= uspec.s, dccOrder = c(1,1), distribution = 'mvt')
fit.s= dccfit(spec.s, data= sres,  fit= multf.s)
show(fit.s)
```

#DCC tgarch mvt AIC= -36.356, BIC= -36.243, Shibata=-36.356, Quinn=-36.314

```
uspec.t= multispec(replicate(4, ugarchspec(variance.model = list(model = "fGARCH", garchOrder = c(1, 1),
submodel = "TGARCH", external.regressors = NULL, variance.targeting = FALSE), mean.model =
list(armaOrder = c(0, 0), include.mean = TRUE, archm = FALSE, archpow = 1, arfima = FALSE,
external.regressors = NULL, archex = FALSE), distribution.model = "std", start.pars = list(), fixed.pars =
list())))
multf.t= multifit(uspec.t, sres)
show(multf.t)
spec.t= dccspec(uspec= uspec.t, dccOrder = c(1,1), model= c("DCC"),distribution = 'mvt')
fit.t= dccfit(spec.t, data= sres,  fit= multf.t, fit.control = list( eval.se = TRUE))
show(fit.t)
```

 # estraggo matrice covarianze e matrice correlazione

```
cov.e= rcov(fit.e)
cor.e= rcor(fit.e)
```

```
dim(cor.e)
dim(cov.e)
#media e varianza condizionata
dcc.mean.cond=fitted(fit.e)
dcc.sigma.cond=sigma(fit.e)
par(mfrow=c(2,1))
plot(dcc.mean.cond)
plot(dcc.sigma.cond)
dev.off()
# prendo le covarianze a due a due
ecov.bit_eth= cov.e[1,2,]
ecov.bit_ltc= cov.e[1,3,]
ecov.eth_ltc= cov.e[2,3,]
ecov.bit_mon= cov.e[1,4,]
ecov.eth_mon= cov.e[2,4,]
ecov.ltc_mon= cov.e[3,4,]

ecov.bit_eth = as.xts(ecov.bit_eth)
ecov.bit_ltc = as.xts(ecov.bit_ltc)
ecov.eth_ltc = as.xts(ecov.eth_ltc)
ecov.bit_mon = as.xts(ecov.bit_mon)
ecov.eth_mon = as.xts(ecov.eth_mon)
ecov.ltc_mon = as.xts(ecov.ltc_mon)

par(mfrow=c(3,1))

plot(ecov.bit_eth)
plot(ecov.bit_ltc)
plot(ecov.eth_ltc)

par(mfrow=c(3,1))

plot(ecov.bit_mon)
plot(ecov.eth_mon)
plot(ecov.ltc_mon)
```

```
dev.off()
# prendo le correlazioni a due a due
ecor.bit_eth= cor.e[1,2,]
ecor.bit_ltc= cor.e[1,3,]
ecor.eth_ltc= cor.e[2,3,]
ecor.bit_mon= cor.e[1,4,]
ecor.eth_mon= cor.e[2,4,]
ecor.ltc_mon= cor.e[3,4,]

ecor.bit_eth <- as.xts(ecor.bit_eth)
ecor.bit_ltc = as.xts(ecor.bit_ltc)
ecor.eth_ltc = as.xts(ecor.eth_ltc)
ecor.bit_mon = as.xts(ecor.bit_mon)
ecor.eth_mon = as.xts(ecor.eth_mon)
ecor.ltc_mon = as.xts(ecor.ltc_mon)

par(mfrow=c(3,1))

plot(ecor.bit_eth)
plot(ecor.bit_ltc)
plot(ecor.eth_ltc)

par(mfrow=c(3,1))
plot(ecor.bit_mon)
plot(ecor.eth_mon)
plot(ecor.ltc_mon)

dev.off()

#Forecast DCC
dcc.for.e = dccforecast(fit.e, n.ahead = 300, n.roll = 0)
forecastmean.e = as.data.frame(fitted(dcc.for.e))
forecastvariancecovariance.e = rcov(dcc.for.e)
cov.for.e= forecastvariancecovariance.e[["2019-12-01"]]
```

```
forecastmean.e = t(forecastmean.e[300,,])
forecastvariancecovariance.e = apply(simplify2array(forecastvariancecovariance.e), c(1,2), mean)

forecastmean.e_yearly = forecastmean.e * 252
forecastvariancecovariance.e_yearly = (forecastvariancecovariance.e) * sqrt(252)

show(dcc.for.e)
plot(dcc.for.e)

# grafici covarianza a due a due
covfor.bit_eth= cov.for.e[1,2,]
covfor.bit_ltc= cov.for.e[1,3,]
covfor.eth_ltc= cov.for.e[2,3,]
covfor.bit_mon= cov.for.e[1,4,]
covfor.eth_mon= cov.for.e[2,4,]
covfor.ltc_mon= cov.for.e[3,4,]

par(mfrow=c(3,1))

plot(covfor.bit_eth)
plot(covfor.bit_ltc)
plot(covfor.eth_ltc)

par(mfrow=c(3,1))

plot(covfor.bit_mon)
plot(covfor.eth_mon)
plot(covfor.ltc_mon)

dev.off()

# grafici correlazione a due a due
cor.for.e= rcor(dcc.for.e)
cor.for.e=cor.for.e[["2019-12-01"]]
ecorfor.bit_eth= cor.for.e[1,2,]
```

```
ecorfor.bit_ltc= cor.for.e[1,3,]
ecorfor.eth_ltc= cor.for.e[2,3,]
ecorfor.bit_mon= cor.for.e[1,4,]
ecorfor.eth_mon= cor.for.e[2,4,]
ecorfor.ltc_mon= cor.for.e[3,4,]

par(mfrow=c(3,1))

plot(ecorfor.bit_eth)
plot(ecorfor.bit_ltc)
plot(ecorfor.eth_ltc)

par(mfrow=c(3,1))

plot(ecorfor.bit_mon)
plot(ecorfor.eth_mon)
plot(ecorfor.ltc_mon)

dev.off()
```

Appendice A.3: Codice R studio – Costruzione portafoglio

```
# Frontiera efficiente
install.packages("parma");
library("parma").

# Creazione specificazione portafoglio con vincoli sui pesi
portfoliospecmult = parmaspec(S = forecastvariancecovariance.e_yearly, forecast = forecastmean.e_yearly,
risk = "EV", riskType = "minrisk", target = mean(forecastmean.e_yearly), targetType = 'equality',  LB =
rep(0.02, 4), UB = rep(0.5, 4), budget = 1, asset.names = sres)
str(portfoliospecmult)
show(portfoliospecmult)

# Risoluzione problema con QP (quadratic) e SOCP (second order cone programming)
portfoliosolveQP = parmasolve(portfoliospecmult, type = 'QP')
portfoliosolveSOCP = parmasolve(portfoliospecmult, type = 'SOCP')
```

```
str(portfoliosolveQP)
str(portfoliosolveSOCP)
show(portfoliosolveQP)
show(portfoliosolveSOCP)

# Estrazione dalla soluzione dei pesi, rendimenti attesi, rischi attesi e serie delle criptovalute
weight = cbind(portfoliosolveQP@solution$weights, portfoliosolveSOCP@solution$weights)
expected_return = cbind(portfoliosolveQP@solution$reward, portfoliosolveSOCP@solution$reward)
expected_risk = cbind(portfoliosolveQP@solution$risk, portfoliosolveSOCP@solution$risk)
criptovalute = portfoliosolveQP@model$asset.names

# Summary
QP_SOCP = cbind('','QP', 'SOCP')
return = cbind('Expected_return ', expected_return )
risk = cbind('Expected_risk', expected_risk)
summary = rbind(QP_SOCP, return, risk)

# Frontiera efficiente
efficientfrontier_QP = parmafrontier(portfoliospecmult, n.points = 100, type = 'QP')
efficientfrontier_SOCP = parmafrontier(portfoliospecmult, n.points = 100, type = 'SOCP')
risk_QP = efficientfrontier_QP[, 'EV']
return_QP = efficientfrontier_QP[, 'reward']
risk_SOCP = efficientfrontier_SOCP[, 'EV']
return_SOCP = efficientfrontier_SOCP[, 'reward']

# Grafico frontiera efficiente
par(mfrow=c(1,1))
plot(risk_QP, return_QP, type = 'p', col = 'steelblue', xlab = 'Expected risk', ylab = 'Expected return', main =
'Efficient frontier', ylim=c(0.01, 0.2), xlim=c(0.000001,0.00013))
lines(risk_SOCP, return_SOCP, col = 'tomato3')
points(parmarisk(portfoliosolveQP), parmareward(portfoliosolveQP), col = 'yellow', pch = 6, lwd = 3)
legend('bottomright', c('Efficient frontier QP', 'Efficient frontier SOCP',  paste('Optimal portfolio (',
round(parmarisk(portfoliosolveQP)/parmareward(portfoliosolveQP), 4), ')', sep = '')), col = c('steelblue',
'tomato1', 'yellow'), pch = c(1, -1, 4, 6), lwd = c(1, 1, 3, 3),
      lty = c(0, 1, 0, 0), bty = 'n', cex = 0.9)
```

```
grid()
```

 # Creazione specificazione portafoglio senza vincoli sui pesi

```
portfoliospecmult = parmaspec(S = forecastvariancecovariance.e_yearly, forecast = forecastmean.e_yearly,
risk = "EV", riskType = "minrisk", target = mean(forecastmean.e_yearly), targetType = 'equality',  LB =
rep(0, 4), UB = rep(1, 4), budget = 1, asset.names = sres)
str(portfoliospecmult)
show(portfoliospecmult)
```

 # Risoluzione problema con QP (quadratic) e SOCP (second order cone programming)

```
portfoliosolveQP = parmasolve(portfoliospecmult, type = 'QP')
portfoliosolveSOCP = parmasolve(portfoliospecmult, type = 'SOCP')
str(portfoliosolveQP)
str(portfoliosolveSOCP)
show(portfoliosolveQP)
show(portfoliosolveSOCP)
```

 # Estrazione dalla soluzione dei pesi, rendimenti attesi, rischi attesi e serie delle criptovalute

```
weight = cbind(portfoliosolveQP@solution$weights, portfoliosolveSOCP@solution$weights)
expected_return = cbind(portfoliosolveQP@solution$reward, portfoliosolveSOCP@solution$reward)
expected_risk = cbind(portfoliosolveQP@solution$risk, portfoliosolveSOCP@solution$risk)
criptovalute = portfoliosolveQP@model$asset.names
```

 # Summary

```
QP_SOCP = cbind(','QP', 'SOCP')
return = cbind('Expected_return ', expected_return )
risk = cbind('Expected_risk', expected_risk)
summary = rbind(QP_SOCP, return, risk)
```

 # Frontiera efficiente

```
efficientfrontier_QP = parmafrontier(portfoliospecmult, n.points = 100, type = 'QP')
efficientfrontier_SOCP = parmafrontier(portfoliospecmult, n.points = 100, type = 'SOCP')
risk_QP = efficientfrontier_QP[, 'EV']
return_QP = efficientfrontier_QP[, 'reward']
```

```
risk_SOCP = efficientfrontier_SOCP[, 'EV']
return_SOCP = efficientfrontier_SOCP[, 'reward']

# Grafico frontiera efficiente
par(mfrow=c(1,1))
plot(risk_QP, return_QP, type = 'p', col = 'steelblue', xlab = 'Expected risk', ylab = 'Expected return', main =
'Efficient frontier', ylim=c(0.01, 0.20), xlim=c(0.000000001,0.00013))
lines(risk_SOCP, return_SOCP, col = 'tomato3')
points(parmarisk(portfoliosolveQP), parmareward(portfoliosolveQP), col = 'yellow', pch = 6, lwd = 3)
legend('bottomright', c('Efficient frontier QP', 'Efficient frontier SOCP',   paste('Optimal portfolio (',
round(parmarisk(portfoliosolveQP)/parmareward(portfoliosolveQP), 4), ')', sep = '')), col = c('steelblue',
'tomato1', 'yellow'), pch = c(1, -1, 4, 6), lwd = c(1, 1, 3, 3),
      lty = c(0, 1, 0, 0), bty = 'n', cex = 0.9)
grid()
```

Appendice A.4: Codice R studio - Calcolo VaR e backtesting

Value at risk

```
install.packages("PerformanceAnalytics");
library("PerformanceAnalytics");

#Historical simulation
weightQP= weight[,1]
weightSOCP= weight[,2]
portfolioreturnQP= Return.portfolio(rX, weights = weightQP)
Var99QP= VaR(portfolioreturnQP, p=0.99, method= "historical")## -0.133
Var95QP= VaR(portfolioreturnQP, p=0.95, method ="historical")##-0.0882
portfolioreturnSOCP=Return.portfolio(rX, weights= weightSOCP)
Var99SOCP= VaR(portfolioreturnSOCP, p=0.99, method="historical")##-0.133
Var95SOCP= VaR(portfolioreturnSOCP, p=0.95, method="historical")##-0.0882


# Backtesting Historical Simulation X=0,95
# estimation window and testing window
osservazioni = nrow(portfolioreturnQP)
```

```
estimation.window = 1078
testing.window = osservazioni - estimation.window
alpha = 0.05

# Backtesting

backTestVaR <- function(x, p = 0.95) {
  historical.VaR = as.numeric(VaR(x, p=p, method="historical"))
  ans = c(historical.VaR)
  names(ans) = c( "HS")
  return(ans)
}

VaR.risultati = rollapply(as.zoo(portfolioreturnQP), width=estimation.window,
                  FUN = backTestVaR, p=0.95, by.column = FALSE,
                  align = "right")
VaR.risultati = lag(VaR.results, k= -1)
chart.TimeSeries(merge(portfolioreturnQP, VaR.risultati), legend.loc="topright")

matrice.violazioni = matrix(0, 1, 5)
rownames(matrice.violazioni) = c( "HS")
colnames(matrice.violazioni) = c("violazioni attese", "violazioni effettive", "1-alpha", "Percentuale", "VR")
matrice.violazioni[, "violazioni attese"] = alpha*testing.window
matrice.violazioni[, "1-alpha"] =  alpha

# Historical simulation violazioni
HSVaR.violazioni = as.zoo(portfolioreturnQP[index(VaR.risultati), ]) < VaR.risultati[, "HS"]
violazioni.date = index(HSVaR.violazioni[which(HSVaR.violazioni)])

# Grafici violazioni
plot(as.zoo(portfolioreturnQP[index(VaR.risultati),]), col="blue", ylab="Return")
abline(h=0)
lines(VaR.risultati[, "HS"], col="black", lwd=2)
lines(as.zoo(portfolioreturnQP[violazioni.date,]), type="p", pch=16, col="red", lwd=2)

#Risultati
```

```
  VaR.violazioni = as.zoo(portfolioreturnQP[index(VaR.risultati), ]) < VaR.risultati[, "HS"]
  matrice.violazioni["HS", "violazioni effettive"] = sum(VaR.violazioni)
  matrice.violazioni["HS", "Percentuale"] = sum(VaR.violazioni)/w.t
  matrice.violazioni["HS", "VR"] = matrice.violazioni["HS", "violazioni effettive"]/matrice.violazioni["HS",
"violazioni attese"]
matrice.violazioni

#Test di Kupiec e di Christoffersen
VaR.test = VaRTest(alpha,
                   actual=coredata(portfolioreturnQP[index(VaR.risultati),]),
                   VaR=coredata(VaR.risultati[,"HS"]))
names(VaR.test)
# Kupiec
VaR.test[1:7]

# Christoffersen
VaR.test[8:12]
# Backtesting Historical Simulation X=0,99
# estimation window and testing window
osservazioni = nrow(portfolioreturnQP)
estimation.window = 1078
testing.window = osservazioni - estimation.window
alpha = 0.01

# Backtesting

backTestVaR <- function(x, p = 0.99) {
  historical.VaR = as.numeric(VaR(x, p=p, method="historical"))
  ans = c(historical.VaR)
  names(ans) = c( "HS")
  return(ans)
}

VaR.risultati = rollapply(as.zoo(portfolioreturnQP), width=estimation.window,
                          FUN = backTestVaR, p=0.99, by.column = FALSE,
```

```
                    align = "right")
VaR.risultati = lag(VaR.results, k= -1)
chart.TimeSeries(merge(portfolioreturnQP, VaR.risultati), legend.loc="topright")

matrice.violazioni = matrix(0, 1, 5)
rownames(matrice.violazioni) = c( "HS")
colnames(matrice.violazioni) = c("violazioni attese", "violazioni effettive", "1-alpha", "Percentuale", "VR")
matrice.violazioni[, "violazioni attese"] = alpha*testing.window
matrice.violazioni[, "1-alpha"] =  alpha

# Historical simulation violazioni
HSVaR.violazioni = as.zoo(portfolioreturnQP[index(VaR.risultati), ]) < VaR.risultati[, "HS"]
violazioni.date = index(HSVaR.violazioni[which(HSVaR.violazioni)])

# Grafici violazioni
plot(as.zoo(portfolioreturnQP[index(VaR.risultati),]), col="blue", ylab="Return")
abline(h=0)
lines(VaR.risultati[, "HS"], col="black", lwd=2)
lines(as.zoo(portfolioreturnQP[violazioni.date,]), type="p", pch=16, col="red", lwd=2)

#Risultati
  VaR.violazioni = as.zoo(portfolioreturnQP[index(VaR.risultati), ]) < VaR.risultati[, "HS"]
  matrice.violazioni["HS", "violazioni effettive"] = sum(VaR.violazioni)
  matrice.violazioni["HS", "Percentuale"] = sum(VaR.violazioni)/w.t
  matrice.violazioni["HS", "VR"] = matrice.violazioni["HS", "violazioni effettive"]/matrice.violazioni["HS",
"violazioni attese"]
matrice.violazioni

#Test di Kupiec e di Christoffersen
VaR.test = VaRTest(alpha,
                   actual=coredata(portfolioreturnQP[index(VaR.risultati),]),
                   VaR=coredata(VaR.risultati[,"HS"]))
names(VaR.test)
# Kupiec
VaR.test[1:7]
```

 # Christoffersen

VaR.test[8:12]

 # VaR con modelli GARCH

 # Calcolo valore critico distribuzione t di student

fitdist(distribution = 'std' , x = portfolioreturnQP)$pars

quantilestd001= qdist(distribution = "std", shape= 2.559362909, p= 0.01)

quantilestd005= qdist(distribution = "std", shape= 2.559362909, p= 0.05)

#EGARCH std AIC= -3.2685, BIC= -3.2481, Shibata=-3.2686, Hannan-Quinn= -3.2610

fit.Stdespecport <- ugarchspec(variance.model = list(model = "eGARCH", garchOrder = c(1, 1), submodel = NULL, external.regressors = NULL, variance.targeting = FALSE), mean.model = list(armaOrder = c(0, 0), include.mean = TRUE, archm = FALSE, archpow = 1, arfima = FALSE, external.regressors = NULL, archex = FALSE), distribution.model = "std", start.pars = list(), fixed.pars = list())

fiteGARCHStdport = ugarchfit(data = portfolioreturnQP, spec = fit.Stdespecport, solver= "hybrid")

show(fiteGARCHStdport)

 # QQ plot

plot(fiteGARCHStdport, which= 9)

#VaR

forc.e = ugarchforecast(fiteGARCHStdport, data = NULL, n.ahead = 1, n.roll = 0, out.sample = 0)

VaR95.e= forc.e@forecast[["sigmaFor"]] * quantilestd005

VaR99.e=forc.e@forecast[["sigmaFor"]] * quantilestd001

#GJR GARCH std AIC= -3.2670, BIC= -3.2466, Shibata=-3.2671, Hannan-Quinn= -3.2594

fit.stdGJRspecport <- ugarchspec(variance.model = list(model = "gjrGARCH", garchOrder = c(1, 1), submodel = NULL, external.regressors = NULL, variance.targeting = FALSE), mean.model = list(armaOrder = c(0, 0), include.mean = TRUE, archm = FALSE, archpow = 1, arfima = FALSE, external.regressors = NULL, archex = FALSE), distribution.model = "std", start.pars = list(), fixed.pars = list())

```
fitGJRGARCHstdport = ugarchfit(data = portfolioreturnQP, spec = fit.stdGJRspecport, solver="hybrid")
show(fitGJRGARCHstdport)

# QQ plot
plot(fitGJRGARCHstdport, which= 9)

#VaR
forc.gjr = ugarchforecast(fitGJRGARCHstdport, data = NULL, n.ahead = 1, n.roll = 0, out.sample = 0)
VaR95.gjr= forc.gjr@forecast[["sigmaFor"]] * quantilestd005
VaR99.gjr=forc.gjr@forecast[["sigmaFor"]] * quantilestd001


#TGARCH std AIC= -3.2667, BIC= -3.2464, Shibata=-3.2668, Hannan-Quinn= -3.2592
fit.stdTspecport <- ugarchspec(variance.model = list(model = "fGARCH", garchOrder = c(1, 1), submodel =
"TGARCH", external.regressors = NULL, variance.targeting = FALSE), mean.model = list(armaOrder =
c(0, 0), include.mean = TRUE, archm = FALSE, archpow = 1, arfima = FALSE, external.regressors =
NULL, archex = FALSE), distribution.model = "std", start.pars = list(), fixed.pars = list())

fitTGARCHstdport = ugarchfit(data = portfolioreturnQP, spec = fit.stdTspecport, solver="hybrid")
show(fitTGARCHstdport)

#QQ plot
plot(fitTGARCHstdport, which= 9)

#VaR
forc.t = ugarchforecast(fitTGARCHstdport, data = NULL, n.ahead = 1, n.roll = 0, out.sample = 0)
VaR95.t= forc.t@forecast[["sigmaFor"]] * quantilestd005
VaR99.t=forc.t@forecast[["sigmaFor"]] * quantilestd001


#SGARCH std AIC= -3.2673, BIC= -3.2503, Shibata=-3.2674, Hannan-Quinn= -3.2610
fit.Stdspecport= ugarchspec(variance.model = list(model = "sGARCH", garchOrder = c(1,1), submodel =
NULL, external.regressors= NULL, variance.targeting = FALSE), mean.model = list(armaOrder = c(0,0),
include.mean = TRUE, archm = FALSE, archpow = 1, arfima = FALSE, external.regressors = NULL,
archex = FALSE), distribution.model = "std", start.pars = list(), fixed.pars = list())
```

```
fit.GARCHStdport = ugarchfit(data = portfolioreturnQP, spec = fit.Stdspecport, solver="hybrid")
show(fit.GARCHStdport)

#QQ plot
plot(fit.GARCHStdport, which= 9)

#VaR
forc.s = ugarchforecast(fit.GARCHStdport, data = NULL, n.ahead = 1, n.roll = 0, out.sample = 0)
VaR95.s= forc.s@forecast[["sigmaFor"]] * quantilestd005
VaR99.s=forc.s@forecast[["sigmaFor"]] * quantilestd001


#IGARCH std AIC= -3.2687, BIC= -3.2551, Shibata=-3.2687, Hannan-Quinn= -3.2636
fit.stdispecport <- ugarchspec(variance.model = list(model = "iGARCH", garchOrder = c(1, 1), submodel =
NULL, external.regressors = NULL, variance.targeting = FALSE), mean.model = list(armaOrder = c(0, 0),
include.mean = TRUE, archm = FALSE, archpow = 1, arfima = FALSE, external.regressors = NULL,
archex = FALSE), distribution.model = "std", start.pars = list(), fixed.pars = list())

fitIGARCHstdport = ugarchfit(data = portfolioreturnQP, spec = fit.stdispecport, solver ="hybrid")
show(fitIGARCHstdport)

#QQ plot
plot(fitIGARCHstdport, which=9)

#VaR
forc.i = ugarchforecast(fitIGARCHstdport, data = NULL, n.ahead = 1, n.roll = 0, out.sample = 0)
VaR95.i= forc.i@forecast[["sigmaFor"]] * quantilestd005
VaR99.i=forc.i@forecast[["sigmaFor"]] * quantilestd001


#Backtesting
#EGARCH
```

forc.rolling.e = ugarchroll(spec = fit.Stdespecport, data = portfolioreturnQP, n.ahead = 1, forecast.length = 500, n.start = NULL, refit.every = 50, refit.window = c("recursive"), window.size = NULL, solver = "hybrid", calculate.VaR = TRUE, VaR.alpha = c(0.01, 0.05), cluster = NULL, keep.coef = TRUE)

show(forc.rolling.e)

#Violation ratio, Kupiec e Christoffersen

report(forc.rolling.e, type = "VaR", VaR.alpha = 0.01, conf.level = 0.99)

report(forc.rolling.e, type = "VaR", VaR.alpha = 0.05, conf.level = 0.95)

 # GJR-GARCH

forc.rolling.gjr = ugarchroll(spec = fit.stdGJRspecport, data = portfolioreturnQP, n.ahead = 1, forecast.length = 500, n.start = NULL, refit.every = 50, refit.window = c("recursive"), window.size = NULL, solver = "hybrid", calculate.VaR = TRUE, VaR.alpha = c(0.01, 0.05), cluster = NULL, keep.coef = TRUE)

show(forc.rolling.gjr)

#Violation ratio, Kupiec e Christoffersen

report(forc.rolling.gjr, type = "VaR", VaR.alpha = 0.01, conf.level = 0.99)

report(forc.rolling.gjr, type = "VaR", VaR.alpha = 0.05, conf.level = 0.95)

 # TGARCH

forc.rolling.t = ugarchroll(spec = fit.stdTspecport, data = portfolioreturnQP, n.ahead = 1, forecast.length = 500, n.start = NULL, refit.every = 50, refit.window = c("recursive"), window.size = NULL, solver = "hybrid", calculate.VaR = TRUE, VaR.alpha = c(0.01, 0.05), cluster = NULL, keep.coef = TRUE)

show(forc.rolling.t)

#Violation ratio, Kupiec e Christoffersen

report(forc.rolling.t, type = "VaR", VaR.alpha = 0.01, conf.level = 0.99)

report(forc.rolling.t, type = "VaR", VaR.alpha = 0.05, conf.level = 0.95)

#SGARCH

forc.rolling.s = ugarchroll(spec = fit.Stdspecport, data = portfolioreturnQP, n.ahead = 1, forecast.length = 500, n.start = NULL, refit.every = 50, refit.window = c("recursive"), window.size = NULL, solver = "hybrid", calculate.VaR = TRUE, VaR.alpha = c(0.01, 0.05), cluster = NULL, keep.coef = TRUE)

```
show(forc.rolling.s)

#Violation ratio, Kupiec e Christoffersen
report(forc.rolling.s, type = "VaR", VaR.alpha = 0.01, conf.level = 0.99)
report(forc.rolling.s, type = "VaR", VaR.alpha = 0.05, conf.level = 0.95)

#IGARCH
forc.rolling.i = ugarchroll(spec = fit.stdispecport, data = portfolioreturnQP, n.ahead = 1, forecast.length =
500, n.start = NULL, refit.every = 50, refit.window = c("recursive"), window.size = NULL, solver =
"hybrid", calculate.VaR = TRUE, VaR.alpha = c(0.01, 0.05), cluster = NULL, keep.coef = TRUE)

show(forc.rolling.i)

#Violation ratio, Kupiec e Christoffersen
report(forc.rolling.i, type = "VaR", VaR.alpha = 0.01, conf.level = 0.99)
report(forc.rolling.i, type = "VaR", VaR.alpha = 0.05, conf.level = 0.95)
```

Riassunto

La presente tesi ha come oggetto l'analisi univariata e multivariata di quattro criptovalute (Bitcoin, Litecoin, Ethereum e Monero) con l'utilizzo di modelli GARCH, al fine di costruirne un portafoglio calcolandone frontiera efficiente e value at risk. Da questo elaborato è possibile capire come sia importante modellare la volatilità per tutte le attività come ad esempio in *asset pricing, portfolio optimization, option pricing, hedging and risk management*. Soprattutto per quanto riguarda l'ultima che, dopo la crisi finanziaria globale del 2008, il quadro normativo internazionale di Basilea III ha imposto dei requisiti patrimoniali più rigorosi e sono stati sviluppati sistemi rafforzati di gestione del rischio. Inoltre, negli ultimi anni l'attenzione degli investitori o comunque del sistema finanziario internazionale si è concentrata nell'affrontare una nuova sfida, ovvero l'introduzione di criptovalute decentrate, il primo è Bitcoin, che è stato creato nel 2009. A differenza delle valute tradizionali, le criptovalute si basano sulla prova crittografica, che fornisce molti vantaggi rispetto ai metodi di pagamento tradizionali (come le carte di credito) tra cui alta liquidità, minori costi di transazione, e l'anonimato. L'interesse per Bitcoin e altre criptovalute è aumentato notevolmente negli ultimi anni. La loro capitalizzazione di mercato è aumentata da circa 18 miliardi di dollari americani all'inizio del 2017 a quasi 600 miliardi alla fine di quell'anno, e gli alti rendimenti hanno attirato nuovi investitori. Inoltre, due grandi borse, ad es. il Chicago Mercantile Exchange (CME) e il Chicago Board Options Exchange (CBOE), hanno iniziato a commerciare futures su Bitcoin. A seguito di questi sviluppi, le banche centrali hanno affrontato la questione se le criptovalute debbano o meno essere regolamentate, date le numerose questioni tecniche e giuridiche coinvolte. La letteratura sulle criptovalute è stata inizialmente dominata da studi sulla sicurezza, aspetti etici e legali di Bitcoin. Invece, la letteratura recente ha esaminato le criptovalute da un punto di vista economico. Tuttavia, poco è noto circa il comportamento dei rendimenti e della volatilità delle criptovalute. Un aspetto particolarmente interessante è se i prezzi altamente volatili delle criptovalute evolvono in modo casuale nel tempo o mostrano una certa prevedibilità. Come viene ricordato in Zouheir Mighri et al. (2019), ci sono diversi risultati che sono stati ottenuti in diversi studi che riporto qui di seguito. Come sostenuto da Cheah e Fry (2015), se il bitcoin fosse una vera unità di conto, o una forma di riserva di valore, non mostrerebbe tale volatilità espressa da bolle e crash. Secondo Dwyer (2015), la media della volatilità mensile del bitcoin è superiore a quella dell'oro o di un insieme di valute estere. Inoltre, trova che la più bassa volatilità mensile per il bitcoin è inferiore alla più alta volatilità mensile per oro e valute. Inoltre, Brière et al. (2015) mostrano che il bitcoin offre significativi benefici di diversificazione per gli investitori, mentre Urquhart (2016) mostra che i rendimenti di bitcoin non seguono una *random walk*. Utilizzando un modello asimmetrico GARCH, Dyhrberg (2016a) dimostra che il bitcoin può essere utile nella gestione del rischio e ideale per gli investitori avversi al rischio in previsione di shock negativi al mercato. Inoltre, Dyhrberg (2016 a, b) mostra che il Bitcoin ha capacità di copertura simili a quelle dell'oro e del dollaro, e come tali può essere impiegato per la gestione del rischio. Inoltre, Balcilar et al. (2017) mostrano che il volume di scambi del bitcoin può prevedere i rendimenti tranne nei regimi di mercato *bull* e *bear* e che il volume non può prevedere la volatilità dei rendimenti bitcoin.

Questa tesi è composta da tre capitoli:

- il primo capitolo è dedicato interamente alla letteratura dei modelli GARCH univariati e multivariati e del loro processo di stima;
- il secondo capitolo presenta, in primo luogo, l'analisi del momento secondo del Bitcoin attraverso l'utilizzo di GARCH univariati e con tre diverse distribuzioni (t di student, GED, NIG). In secondo luogo, l'analisi si sposterà in campo multivariato analizzando insieme quattro criptovalute (Bitcoin, Litecoin, Ethereum e Monero). Il modello scelto per effettuare tale analisi è il DCC-GARCH (par. 1.5.6), il quale ha il vantaggio di avere un processo di stima in due passaggi, il che semplifica molto tale processo. Nella prima fase si stimano *m* modelli GARCH univariati (possono essere anche diversi modelli per ognuna delle serie storiche) per ognuna delle serie storiche considerate e vengono così costruite le matrici D_t (vd. par. 1.5.6) dalle quali si ricavano i residui standardizzati. Il secondo passaggio è quello appunto di calcolare le correlazioni dinamiche dei residui standardizzati ricavati in precedenza;
- il terzo capitolo è invece dedicato alla creazione di un portafoglio con le quattro criptovalute analizzate. Il quale verrà ottimizzato costruendo la frontiera efficiente attraverso la soluzione del problema di minimizzazione del rischio (con l'aiuto del pacchetto "parma" su R). Infine, verrà stimato il VaR della serie di rendimenti del portafoglio e verrà verificata anche la bontà di tale stima. A tale scopo, è analizzato il *forecast* (con un orizzonte temporale di 300 periodi) del modello DCC stimato nel par. 2.6. Ciò che interessa a noi per il calcolo della frontiera efficiente sono i rendimenti attesi e le varianze e covarianze attese. I forecast dei rendimenti e delle varianze/covarianze sono oggetti di dimensioni, rispettivamente, (300 x 4) e (4 x 4 x 300). Di questi, calcoliamo i valori medi e li annualizziamo in quanto valori giornalieri.

Quindi, l'analisi svolta può essere divisa in quattro parti. Nella prima parte, svolgiamo un'analisi del Bitcoin modellizzando la sua volatilità condizionata attraverso diversi GARCH univariati e con tre diverse distribuzioni. Come distribuzioni utilizziamo la t di student, la GED, e la normale inversa gaussiana per catturare l'eccesso di curtosi e di skewness presente nella serie dei rendimenti. Il modello scelto è un TGARCH con distribuzione normale inversa gaussiana nonostante non sia quello con Akaike, BIC, Shibata e Hannan-Quinn più basso. Questo perché è l'unico modello, insieme al GARCH integrato con distribuzione GED, che non presenta autocorrelazioni seriali nei residui standardizzati e nei residui al quadrato. Infine, il TGARCH è preferito rispetto all'IGARCH in quanto presenta più parametri statisticamente significativi. La seconda parte dell'analisi si svolge in campo multivariato, analizzando quattro criptovalute: Bitcoin, Ethereum, Litecoin e Monero. Scegliamo un modello DCC-GARCH per la semplicità del processo di stima diviso in due fasi. La prima fase di stima di GARCH univariati e la seconda fase consiste invece nella stima della matrice di correlazione R_t. La stima del modello è in figura 21. I modelli GARCH univariati, utilizzati per la prima fase del processo di stima, sono gli stessi usati nel caso dell'analisi del Bitcoin, ovvero SGARCH, TGARCH, GJR-GARCH, IGARCH e EGARCH. Però in questo caso viene utilizzata solo la distribuzione t di student. I valori

di Akaike e degli altri criteri sono molto simili tra di loro, ma il modello EGARCH è quello che presenta più parametri statisticamente significativi. Inoltre, il modello EGARCH riesce a catturare anche l'effetto leverage che è presente nelle serie. I parametri del DCC sono anch'essi statisticamente significativi e quindi l'EGARCH viene scelto come modello migliore. Nell'analisi della correlazione, vediamo che le varie criptovalute prese a due a due sono molto correlate tra di loro. Però questa correlazione è poco accentuata nel periodo 2015-2017 (assumendo addirittura valori negativi anche se solo per pochi giorni), ma da metà-fine anno 2017 la correlazione cresce esponenzialmente fino a toccare valori vicino all'unità. Questo è dovuto forse al fatto che nel 2017 è cresciuto molto l'interesse degli investitori nei confronti del Bitcoin (c'è stato il boom del prezzo del Bitcoin che nel corso del 2017 ha raggiunto valori intorno ai 20000 $) e delle criptovalute in generale e ciò ha fatto crescere la correlazione fra le varie criptovalute. Nella terza parte dell'analisi, grazie alle stime dei rendimenti attesi e delle covarianze e varianze attese ottenute con i forecast del modello DCC-GARCH, costruiamo il nostro portafoglio di criptovalute composto naturalmente da Bitcoin, Ethereum, Litecoin e Monero. Un portafoglio di investimento è un investimento passivo in titoli, per cui non è necessaria una gestione attiva dei titoli da parte dell'investitore. Può essere anche visto come un investimento fatto da un investitore, il quale non è particolarmente interessato nella gestione della società in cui sta investendo. Quindi, il principale scopo di un investimento in un portafoglio di titoli è il guadagno finanziario che ne deriva. Ogni investitore vorrebbe ottenere il maggior profitto possibile da un investimento, ma questo è controbilanciato da un certo ammontare di rischio che gli investitori sono capaci o comunque sono disposti a prendere con sé. Il rendimento atteso e il rischio misurato dalla varianza (o dalla standard deviation, ovvero la radice quadrata della varianza) sono le due caratteristiche principali di un portafoglio di investimento. Purtroppo, degli studi mostrano come gli investimenti che generano dei rendimenti maggiori siano anche quelli più rischiosi. In questa sezione utilizzeremo la frontiera efficiente per indicare il portafoglio ottimale. La frontiera efficiente è l'insieme dei portafogli ottimali che offrono il rendimento atteso più elevato per un determinato livello di rischio o il rischio più basso per un determinato livello di rendimento atteso. I portafogli che si trovano al di sotto della frontiera efficiente non sono ottimali perché non forniscono un rendimento sufficiente per il livello di rischio. I portafogli che raggruppano a destra della frontiera efficiente sono non ottimali perché presentano un livello più elevato di rischio per il tasso di rendimento definito. La frontiera efficiente fu introdotta dal premio Nobel per l'economia Harry Markowitz nel 1952.

Con l'aiuto del pacchetto parma (portfolio allocation and risk management application) su R, riusciamo a calcolare i pesi ottimali del nostro portafoglio, rappresentando la nostra frontiera efficiente (figura 30). Notiamo, inoltre, che se non imponiamo vincoli ai pesi nel processo di ottimizzazione, il portafoglio che si ottiene (portafoglio ottimale non vincolato) è preferibile a quello vincolato. Tuttavia, senza l'utilizzo di vincoli è possibile che vengano prodotti dei portafogli troppo concentrati e questo è molto rischioso. Infatti, dato che non c'è certezza che il modello e le stime siano esatte, la diversificazione è una contromisura per difendersi da eventi non prevedibili. Infine, nella quarta parte dell'analisi, andiamo a stimare il value at risk della serie di rendimenti del portafoglio creato con i pesi ottimali ottenuti in precedenza. Gli economisti finanziari sono

sempre alla ricerca di nuovi modelli per modellizzare la volatilità nei rendimenti degli assets. Infatti, la volatilità è un elemento importante in quanto misura del rischio di un asset e, quindi, se la volatilità è alta gli investitori chiederanno un premio più alto per investire in tali assets assumendosi un rischio maggiore. Tuttavia, la volatilità è una misura del rischio ingannevole in quanto non è detto che in un periodo di bassa volatilità anche il rischio sui mercati finanziari sia basso. Pertanto, banche e istituzioni finanziare applicano i modelli value-at-risk per valutare i rischi che stanno assumendo nello svolgimento delle loro attività. Il value at risk (VaR) è un metodo statistico che calcola un singolo numero per riassumere il rischio complessivo di un portafoglio finanziario. Potrebbe essere anche utilizzato per definire il capitale che una banca deve mantenere rispetto ai rischi che sta assumendo. Quando si utilizza il VaR, si sta affermando che: "Sono X per cento sicuro che l'istituto finanziario non incorrerà una perdita maggiore di V nei prossimi T giorni". La variabile V indica il quantile del VaR del portafoglio. Il value at risk è una funzione con due parametri: il livello di confidenza (X%) e l'orizzonte temporale o il periodo di detenzione. Il VaR potrebbe anche essere definito come il livello di perdita in un periodo di T giorni che ha una probabilità (100-X) % di essere superato. Nelle banche, le autorità di regolamentazione richiedono il calcolo del VaR per i rischi di mercato con un orizzonte temporale di 10 giorni e un livello di confidenza di X = 99 %. Se prendiamo T come periodo di detenzione e X% come livello di confidenza, il VaR è la perdita che corrisponde al (100-X) percentile della distribuzione dei guadagni del portafoglio nei prossimi T giorni. Inoltre, interessante da accennare, se si tiene conto della distribuzione di probabilità del guadagno, il value at risk è relativo alla coda sinistra della distribuzione mentre se consideriamo la distribuzione delle perdite, il value at risk è relativo alla coda destra della distribuzione. Nel primo caso, la perdita è un guadagno negativo, mentre nel secondo il guadagno è una perdita negativa.

Le metodologie utilizzate per la stima del VaR sono due: *historical simulation* e l'utilizzo di GARCH univariati che permettono appunto la stima della varianza condizionata al tempo $t + 1$. Il primo metodo usa osservazioni passate dei rendimenti per effettuare previsioni del VaR. Esso consiste in tre steps:

- ordinare la serie dei rendimenti del portafoglio dal valore più piccolo a quello più grande
- va trovato il p% quantile della serie ordinata. Ad esempio, se $p = 0{,}01$ (in caso di VaR (99%)) allora $q = p * N$ dove N è il numero delle osservazioni. Nel nostro caso, $q = 0{,}01 * 1577 = 15{,}77 \sim 16$.
- estraiamo il q-esimo valore dalla serie ordinata, in questo caso il 16-esimo.

Il secondo metodo consiste ancora in tre steps:

- stima della volatilità utilizzando dei modelli GARCH univariati (applicando tali modelli sulla serie dei rendimenti del portafoglio)
- *forecast* della volatilità utilizzando i modelli di cui sopra (TGARCH, GJR-GARCH, EGARCH, SGARCH, IGARCH)
- calcolo del VaR utilizzando la volatilità condizionata predetta.

Ipotizzando $\mu = 0$, VaR $= \sigma_{t+1} * \mathrm{F}^{-1}(\alpha)$ dove σ_{t+1} è la standard deviation predetta per t+1 condizionata ai valori passati e F^{-1} è la funzione inversa della densità di probabilità della distribuzione t di student. Nella stima dei modelli viene utilizzata la distribuzione t di student.
Inoltre, per verificare la bontà delle previsioni del VaR effettuate dalle due metodologie, utilizziamo dei diversi test statistici: violation ratio, test di Kupiec e test di Christoffersen.

Violation ratio: se le perdite effettive eccedono i forecast del VaR, allora il value at risk è considerate violato. Il rapporto di violazione (violation ratio) è la somma dei casi di superamento diviso per il numero atteso di casi di superamento. Il rapporto è quindi dato da (Danielsson, 2011):

$$VR = \frac{E}{p*N}$$

dove

- E è il numero osservato di casi di superamento
- p è il livello di probabilità del VaR, nel nostro caso 0,05 e 0,01
- N è il numero di osservazioni utilizzate per il forecast del value at risk

Molti risk managers concordano che il valore ottimale di tale rapporto debba trovarsi tra 0,8 e 1,2.

Test di Kupiec: assumiamo che i casi di superamente nel tempo seguano una distribuzione binomiale e l'obiettivo del test di Kupiec è quello di determinare la consistenza di queste violazioni con un dato livello di confidenza. Se il numero di superamenti osservati differisce di molto da quelli attesi, allora l'adeguatezza del modello deve essere verificata. Per effettuare il test, abbiamo bisogno del numero effettivo di violazioni (E), numero di osservazioni utilizzate per il forecast (N) e la probabilità del VaR (p). Assumendo, quindi, che E sia distribuito come una Binomiale (N, p), l'ipotesi nulla è:

$$H_0: p = p_0, \quad p_0 = 0{,}01 \; o \; 0{,}05$$

e p è stimato come $\frac{E}{N}$. Quindi, andiamo a testare se il numero di violazioni osservate sia significativamente differente da quello atteso. Kupiec (1995) propose questo test:

$$-2\ln[(1-p)^{N-E}p^{E}] + 2\ln[(1-p_0)^{N-E}{p_0}^{E}]$$

che segue una distribuzione chi-quadro con un grado di libertà.

Test di Christoffersen: se le variazioni giornaliere del portafoglio fossero indipendenti, le violazioni dovrebbero essere distribuite uniformemente per tutto il periodo utilizzato per i test retrospettivi (backtesting).

In pratica, sono spesso raggruppati suggerendo che le perdite nei giorni successivi non sono indipendenti (John C. Hull, 2015). Un modo per verificare ciò è il test statistico proposto da Christoffersen (1998):

$$-2\ln[(1-\pi)^{u_{00}+u_{10}}\pi^{u_{01}+u_{11}}] + 2\ln\left[(1-\pi_{01})^{u_{00}}\pi_{01}^{u_{01}}(1-\pi_{11})^{u_{10}}\pi_{11}^{u_{11}}\right]$$

dove u_{ij} è il numero di osservazioni in cui si va da un giorno in cui siamo nello stato i ad un giorno in cui siamo nello stato j. Lo stato 0 è il giorno in cui non ci sono violazioni mentre lo stato 1 è il giorno dove si verifica una violazione. Inoltre, π, π_{01}, π_{11} sono definiti come (John C. Hull, 2015):

$$\pi = \frac{u_{01} + u_{11}}{u_{00} + u_{01} + u_{10} + u_{11}}$$

$$\pi_{01} = \frac{u_{01}}{u_{00} + u_{01}}$$

$$\pi_{11} = \frac{u_{11}}{u_{10} + u_{11}}$$

Da questi test ricaviamo che i GARCH univariati sono in grado di fare delle stime più accurate del VaR rispetto al metodo *historical simulation*. In particolare, vediamo che, nel caso di un livello di confidenza del 99%, le stime dei diversi GARCH univariati hanno la stessa bontà previsionale. Inoltre, in questo caso anche il metodo HS è un buon stimatore. Invece, nel caso di un livello di confidenza del 95%, i modelli GARCH sono tutti dei buoni stimatori, però il modello SGARCH è quello che presenta un rapporto di violazione di 1 quindi sembra essere il miglior modello. Invece, per quanto riguarda il metodo HS, dai test statistici ricaviamo che non è un buon modello previsionale.