

Ensayo sobre Modelos de Volatilidad

AUTHOR

Mario Nathaniel de la Vega Ramirez

Volatilidades

La volatilidad es la medida que se hace a la incertidumbre, y la primera aproximación estadística que cuantifica la variabilidad con respecto a su media, o dispersión de un conjunto de datos es la varianza muestral. En este caso, de los rendimientos de un activo financiero a lo largo del tiempo.

Existe una larga historia acerca de la modelación de las series de precios de acciones, y con ello, distintos supuestos sobre la volatilidad. Se puede rastrear que la volatilidad fija en el tiempo está relacionada a la ecuación de Black-Scholes-Merton para la valoración de opciones financieras, y que los precios de los activos siguen un movimiento Browniano Geométrico, que asume que la volatilidad es constante a lo largo del tiempo y que el precio está log-normal distribuido en el tiempo t . Estos supuestos solamente nos ayudan en un primer momento para conseguir el mejor precio o el precio justo de la opción, pero para un modelado más cercano a la realidad se pueden tomar otras alternativas, ya que la prevalencia de los shocks es particularmente alta, ya que estos tienen el mismo peso a lo largo del tiempo.

Según el autor Ruey S. Tsay (2005) en su libro Analysis of Financial Time Series la volatilidad tiene la característica de que no es observable en un primer momento, pues, si tu tienes el log-retorno de un determinado activo ese día, no puedes conseguirla de este solo. También que se pueden encontrar ciertos patrones, como acumulaciones o conglomerados estacionales, donde habrá periodos de alta volatilidad o baja, fenómeno que también suele ser mencionado como memoria de los mercados, finalmente que evoluciona de manera más o menos continua, y que esta no diverge a infinito. Según el autor estas características, estadísticamente hablando, provocan que esta sea comúnmente estacionaria. Y como se ha visto también, esta tiene un efecto apalancamiento, o una reacción asimétrica con respecto a su comportamiento a subida o bajada de precios.

Una primera aproximación para un modelo es el denominado modelo ARCH, de 1982 por Engle, que, es destibar la volatilidad en función de los shocks o innovaciones al cuadrado (errores por su volatilidad en t ,

o desviaciones con respecto a la media). Se supone que no hay correlación serial, pero si dependencia entre cada shock. En este caso la ponderación de los shocks la hacen los mismos datos.

Bollerslev en 1986 realiza una generalización de este modelo, suponiendo que este depende no solo de las inovaciones, sino que depende de la volatilidad en el tiempo anterior, que, concentra en esta ya una cantidad de shocks al cuadrado, que da como resultado, minimizar la cantidad de terminos necesarios para una adecuada estimación. Este es el modelo GARCH. Sin embargo este no considera el efecto apalancamiento, por lo que se desarrollan otros modelos como el EGARCH de Nelson (1991) y el GJR-GARCH de Glosten, Jagannathan y Runkle (1993), que consideran este efecto de asimetria. Por la construcción del modelo, la volatilidad predicha suele capturar mejor los conglomerados de volatilidad.

Algunas de las desventajas de estos modelos es que requieren de muchas observaciones para ser estimados de manera confiable, computacionalmente son más complejos, es más complejo de explicar de manera no técnica, así como que se deben probar distintas alternativas para la distribución del error.

En el caso de RiskMetricsTM, el uso de media móvil ponderada exponencialmente, por sus siglas en ingles, EWMA, es el método que J. P. Morgan expone en 1996. Supuestamente, los motivos por los que lo prefieren son la respuesta rápida ante shocks recientes, y también, tiene la ventaja de que tras un gran shock, esta igual decaera de manera exponencial a medida que el peso del shock cae. Las desventajas de utilizar este método es encontrar un parametro de decaimiento optimo, así como la falta de captura de la asimetría en la volatilidad.

Se puede hacer notar que este es un caso especial de GARCH(1,1), o más explícitamente un IGARCH(1,1).

El factor de decaimiento que riskmetrics, de manera numérica estima como optimo para series diarias es de 0.94.

Algo importante a mencionar, es que pronosticos mencionados son al corto plazo para todos los modelos o supuestos vistos, ya que, el largo plazo para la volatilidad es infinitamente más complejo.

Con esta breve recapitulación de las ventajas y desventajas se busca sostener que el mejor modelo es el que mejor se adapte a los datos en su momento, y que si es necesario, habrá que cambiarlo.

```
TimeFixedSigma = sd(rendimientos$QQQ)
```

```
lambda1 = 0.95
weights1 = lambda1^(0:(length(rendimientos$QQQ)-1))

EWSigma95 = sqrt( sum( weights1 * (rendimientos$QQQ - mean(rendimientos$QQQ))^2 ) * (1 - lambda1) )

lambda2 = 0.98
weights2 = lambda2^(0:(length(rendimientos$QQQ)-1))

EWSigma98 = sqrt( sum( weights2 * (rendimientos$QQQ - mean(rendimientos$QQQ))^2 ) * (1 - lambda2) )

## GARCH

spec <- ugarchspec(variance.model = list(model = "sGARCH",
                                         garchOrder = c(1, 1)),
                  mean.model = list(armaOrder = c(0, 0), include.mean = TRUE),
                  distribution.model = "norm")

norm_garch <- ugarchfit(spec = spec, data = rendimientos$QQQ)

spec <- ugarchspec(variance.model = list(model = "sGARCH",
                                         garchOrder = c(1, 1)),
                  mean.model = list(armaOrder = c(0, 0), include.mean = TRUE),
                  distribution.model = "std")
std_garch <- ugarchfit(spec = spec, data = rendimientos$QQQ)

spec <- ugarchspec(variance.model = list(model = "sGARCH",
                                         garchOrder = c(1, 1)),
                  mean.model = list(armaOrder = c(0, 0), include.mean = TRUE),
                  distribution.model = "ged")
ged_garch <- ugarchfit(spec = spec, data = rendimientos$QQQ)

## GJR-GARCH

spec <- ugarchspec(variance.model = list(model = "gjrGARCH",
```

```

        garchOrder = c(1, 1)),
    mean.model = list(armaOrder = c(0, 0), include.mean = TRUE),
    distribution.model = "norm")
norm_gjrgarch <- ugarchfit(spec = spec, data = rendimientos$QQQ)

spec <- ugarchspec(variance.model = list(model = "gjrGARCH",
        garchOrder = c(1, 1)),
    mean.model = list(armaOrder = c(0, 0), include.mean = TRUE),
    distribution.model = "std")
std_gjrgarch <- ugarchfit(spec = spec, data = rendimientos$QQQ)

spec <- ugarchspec(variance.model = list(model = "gjrGARCH",
        garchOrder = c(1, 1)),
    mean.model = list(armaOrder = c(0, 0), include.mean = TRUE),
    distribution.model = "ged")
ged_gjrgarch <- ugarchfit(spec = spec, data = rendimientos$QQQ)

spec <- ugarchspec(variance.model = list(model = "iGARCH",
        garchOrder = c(1, 1)),
    mean.model = list(armaOrder = c(0, 0), include.mean = TRUE),
    distribution.model = "ged")
EWMA <- ugarchfit(spec = spec, data = rendimientos$QQQ)

AIC.rugarch=function(object){
  k= length(object@fit$coef)
  llf=object@fit$LLH
  aic=(2*k)-(2*llf)
  return(aic)
}

## Para la varianza fija, se hara el tipico supuesto de normalidad y se utilizara
# como función de verosimilitud.

n = length(rendimientos$QQQ)

```

```
mu = mean(rendimientos$QQQ)
sigma2 = var(rendimientos$QQQ)

logLTS = -n/2*log(2*pi) - n/2*log(sigma2) - (1/(2*sigma2))*sum((rendimientos$QQQ - mu)^2)
```

```
AIC_comparativo = c("Varianza Tiempo Fijo ~ Normales" = (2)-2*logLTS,
  "EWMA aproximado" = (2)-2*EWMA@fit$LLH,
  "GARCH err ~ Normales" = AIC.rugarch(norm_garch),
  "GARCH err ~ t" = AIC.rugarch(std_garch),
  "GARCH err ~ GED" = AIC.rugarch(ged_garch),
  "GJR-GARCH err ~ Normales" = AIC.rugarch(norm_gjrgarch),
  "GJR-GARCH err ~ t" = AIC.rugarch(std_gjrgarch),
  "GJR-GARCH err ~ GED" = AIC.rugarch(ged_gjrgarch)
)

cat("El criterio de infomación de Akaike para los modelos en orden descendente son:\n\n")
```

El criterio de infomación de Akaike para los modelos en orden descendente son:

```
sort(AIC_comparativo)
```

EWMA aproximado	Varianza Tiempo Fijo ~ Normales
-1484.458	-1481.139
GJR-GARCH err ~ Normales	GJR-GARCH err ~ t
-1478.739	-1478.702
GJR-GARCH err ~ GED	GARCH err ~ t
-1478.128	-1477.972
GARCH err ~ Normales	GARCH err ~ GED
-1477.212	-1477.059

Se espera que la varianza convencional y el modelo EWMA tenga un criterio de información mas pequeño a priori, y se hizo un aproximación a este. A priori con el criterio de infomrmación de Akaike, el mejor modelo entre los GARCH es el GJR-GARCH con errores normales. Sin embargo el EWMA es incluso más parsimonioso que la varianza a tiempo fijo, pero es solo una primera aproximación.

```
## Volatilidades en rendimientos
```

```
Ventana = 100
```

```
Vol_r = data.frame( Date = rendimientos$Date,
  TFS = rollapply(rendimientos$QQQ, FUN = sd, width = Ventana, fill = NA, al
  EWS95 = rolleWSigma(rendimientos$QQQ, lambda = lambda1 ,ventana = Ventana),
  EWS98 = rolleWSigma(rendimientos$QQQ, lambda = lambda2 ,ventana = Ventana),
  garch.norm = rollGARCH(rendimientos$QQQ, model = "sGARCH", LLF = "norm", ve
  garch.t = rollGARCH(rendimientos$QQQ, model = "sGARCH", LLF = "std", ventar
  garch.ged = rollGARCH(rendimientos$QQQ, model = "sGARCH", LLF = "ged", vent
  gjrgarch.norm = rollGARCH(rendimientos$QQQ, model = "gjrGARCH" , LLF = "nor
  gjrgarch.t = rollGARCH(rendimientos$QQQ, model = "gjrGARCH" , LLF = "norm",
  gjrgarch.ged = rollGARCH(rendimientos$QQQ, model = "gjrGARCH" , LLF = "norm
)
```

```
##
```

```
r.plot= plot_ly() %>%
```

```
  add_trace(data = rendimientos, x = ~rendimientos$Date, y= ~rendimientos$QQQ,
    type = "scatter", mode = 'lines')
```

```
precios.plot= plot_ly() %>%
```

```
  add_trace(data = precios, x = ~precios$Date, y= ~ precios$QQQ,
    type = "scatter", mode = 'lines')
```

```
Sigma.plot.com = plot_ly() %>%
```

```
  add_trace(data = Vol_r, x = ~Vol_r$Date, y = ~Vol_r$EWS95,
    type = "scatter", mode = "lines", name = "Volatilidad SE ( $\lambda=0.95$ )") %>%
```

```
  add_trace(data = Vol_r, x = ~Vol_r$Date, y = ~Vol_r$EWS98,
    type = "scatter", mode = "lines", name = "Volatilidad SE ( $\lambda=0.98$ )") %>%
```

```
  add_trace(data = Vol_r, x = ~Vol_r$Date, y = ~Vol_r$garch.norm,
    type = "scatter", mode = "lines", name = "Volatilidad GARCH(1,1) Normales") %>%
```

```
  add_trace(data = Vol_r, x = ~Vol_r$Date, y = ~Vol_r$garch.t,
    type = "scatter", mode = "lines", name = "Volatilidad GARCH(1,1) t") %>%
```

```
  add_trace(data = Vol_r, x = ~Vol_r$Date, y = ~Vol_r$garch.ged,
```

```

        type = "scatter", mode = "lines", name = "Volatilidad GARCH(1,1) GED") %>%
add_trace(data = Vol_r, x = ~Vol_r$Date, y = ~Vol_r$gjrgarch.norm,
        type = "scatter", mode = "lines", name = "Volatilidad GJR-GARCH(1,1) Normales") %>%
add_trace(data = Vol_r, x = ~Vol_r$Date, y = ~Vol_r$gjrgarch.t,
        type = "scatter", mode = "lines", name = "Volatilidad GJR-GARCH(1,1) t") %>%
add_trace(data = Vol_r, x = ~Vol_r$Date, y = ~Vol_r$gjrgarch.ged,
        type = "scatter", mode = "lines", name = "Volatilidad GJR-GARCH(1,1) GED") %>%
add_trace(data = Vol_r, x = ~Vol_r$Date, y = ~Vol_r$TFS,
        type = "scatter", mode = "lines", name = "Volatilidad de Tiempo Fijo") %>%
layout(title = "Volatilidades Estimadas",
        xaxis = list(title = "Fecha"),
        yaxis = list(title = "Volatilidad"),
        hovermode = "x unified")

TFS.plot = plot_ly() %>%
  add_trace(data = Vol_r, x = ~Vol_r$Date, y = ~Vol_r$TFS,
            type = "scatter", mode = "lines", name = "Volatilidad de Tiempo Fijo", line = list(

EWS95.plot = plot_ly() %>%
  add_trace(data = Vol_r, x = ~Vol_r$Date, y = ~Vol_r$EWS95,
            type = "scatter", mode = "lines", name = "EWS95", line = list(color = '#004400'))

EWS95.plot = plot_ly() %>%
  add_trace(data = Vol_r, x = ~Vol_r$Date, y = ~Vol_r$EWS98,
            type = "scatter", mode = "lines", name = "EWS98", line = list(color = '#004400'))

garch.norm.plot = plot_ly() %>%
  add_trace(data = Vol_r, x = ~Vol_r$Date, y = ~Vol_r$garch.norm,
            type = "scatter", mode = "lines", name = "Volatilidad GARCH(1,1) Normales", line =

garch.t.plot = plot_ly() %>%
  add_trace(data = Vol_r, x = ~Vol_r$Date, y = ~Vol_r$garch.t,
            type = "scatter", mode = "lines", name = "Volatilidad GARCH(1,1) t", line = list(cc

garch.ged.plot = plot_ly() %>%
  add_trace(data = Vol_r, x = ~Vol_r$Date, y = ~Vol_r$garch.ged,
            type = "scatter", mode = "lines", name = "Volatilidad GARCH(1,1) GED", line = list(

```

```

gjrgarch.norm.plot = plot_ly() %>%
  add_trace(data = Vol_r, x = ~Vol_r$Date, y = ~Vol_r$gjrgarch.norm,
            type = "scatter", mode = "lines", name = "Volatilidad GJR-GARCH(1,1) Normales", lin

gjrgarch.t.plot = plot_ly() %>%
  add_trace(data = Vol_r, x = ~Vol_r$Date, y = ~Vol_r$gjrgarch.t,
            type = "scatter", mode = "lines", name = "Volatilidad GJR-GARCH(1,1) t", line = lis

gjrgarch.ged.plot = plot_ly() %>%
  add_trace(data = Vol_r, x = ~Vol_r$Date, y = ~Vol_r$gjrgarch.ged,
            type = "scatter", mode = "lines", name = "Volatilidad GJR-GARCH(1,1) GED", line = 1

```

Evaluación de desempeño real

Ahora se buscara ver que modelo realmente se adapta mejor a los datos reales, para ello se usará la prueba de Kupiec, que evalua el número de violaciones que tiene un modelo de manera teórica mediante un experimento binomial y los realmente sucedidos.

```

tickerV=c("QQQ")
deD="2019-11-20"
hastaD="2024-11-20"
per="D"
paridadFX="USDMXN=X"
convertirFX=c(TRUE)

Datos_lp=historico_multiples_precios(tickers=tickerV,de=deD,hasta=hastaD,periodicidad=per,fxRat

precios_lp=Datos_lp$tablaPrecios
rendimientos_lp=Datos_lp$tablaRendimientosCont
r_lp=diff(log(precios_lp$QQQ),1)
rendimientos_lp$QQQ = r_lp

PL_lp=diff(precios_lp$QQQ,1)

```

```

precios_lp$PL_lp=c(NA,PL_lp)

precios_lp$r_lp=c(NA,r_lp)

r_lp.plot= plot_ly() %>%
  add_trace(data = rendimientos_lp, x = ~rendimientos_lp$Date, y= ~rendimientos_lp$QQQ,
            type = "scatter", mode = 'lines')

precios_lp.plot= plot_ly() %>%
  add_trace(data = precios_lp, x = ~precios_lp$Date, y= ~ precios_lp$QQQ,
            type = "scatter", mode = 'lines')

```

Volatilidades en rendimientos

Ventana = 200

IMPORTANTE: A PESAR DE QUE LA TAREA ESPECIFICA 250 DÍAS SE TOMARAN 200 POR MOTIVOS DE CONVERG

```

Vol_r_lp = data.frame( Date = rendimientos_lp$Date,
  TFS = rollapply(rendimientos_lp$QQQ, FUN = sd, width = Ventana, fill = NA,
  EWS95 = rolleWSigma(rendimientos_lp$QQQ,lambda = lambda1 ,ventana = Ventana
  EWS98 = rolleWSigma(rendimientos_lp$QQQ,lambda = lambda2 ,ventana = Ventana
  garch.norm = rollGARCH(rendimientos_lp$QQQ, model = "sGARCH", LLF = "norm",
  garch.t = rollGARCH(rendimientos_lp$QQQ, model = "sGARCH", LLF = "std", ver
  garch.ged = rollGARCH(rendimientos_lp$QQQ, model = "sGARCH", LLF = "ged", v
  gjrgarch.norm = rollGARCH(rendimientos_lp$QQQ, model = "gjrGARCH" , LLF = '
  gjrgarch.t = rollGARCH(rendimientos_lp$QQQ, model = "gjrGARCH" , LLF = "nor
  gjrgarch.ged = rollGARCH(rendimientos_lp$QQQ, model = "gjrGARCH" , LLF = "r
)

```

Prueba de Kupiec

M = 500000

alpha = 0.05

Se usará CVaR al 95%

```

CVar95.TFS = CVaR(M = M, sigma=Vol_r_lp$TFS, confidence = 1-alpha, pdfFunct = "norm", VaRt = 1)
CVar95.EWS95 = CVaR(M = M, sigma=Vol_r_lp$EWS95, confidence = 1-alpha, pdfFunct = "norm", VaRt = 1)
CVar95.EWS98 = CVaR(M = M, sigma=Vol_r_lp$EWS98, confidence = 1-alpha, pdfFunct = "norm", VaRt = 1)
CVar95.garch.norm = CVaR(M = M, sigma=Vol_r_lp$garch.norm, confidence = 1-alpha, pdfFunct = "norm", VaRt = 1)
CVar95.garch.t = CVaR(M = M, sigma=Vol_r_lp$garch.t, confidence = 1-alpha, pdfFunct = "t", VaRt = 1)
CVar95.garch.ged = CVaR(M = M, sigma=Vol_r_lp$garch.ged, confidence = 1-alpha, pdfFunct = "ged", VaRt = 1)
CVar95.gjrgarch.norm = CVaR(M = M, sigma=Vol_r_lp$gjrgarch.norm, confidence = 1-alpha, pdfFunct = "norm", VaRt = 1)
CVar95.gjrgarch.t = CVaR(M = M, sigma=Vol_r_lp$gjrgarch.t, confidence = 1-alpha, pdfFunct = "t", VaRt = 1)
CVar95.gjrgarch.ged = CVaR(M = M, sigma=Vol_r_lp$gjrgarch.ged, confidence = 1-alpha, pdfFunct = "ged", VaRt = 1)

```

```

TFS.KT =KupiecBackTest(returns = rendimientos_lp$QQQ*M, riskValues = CVar95.TFS, alpha = alpha)
EWS95.KT =KupiecBackTest(returns = rendimientos_lp$QQQ*M, riskValues = CVar95.EWS95, alpha = alpha)
EWS98.KT =KupiecBackTest(returns = rendimientos_lp$QQQ*M, riskValues = CVar95.EWS98, alpha = alpha)
garch.norm.KT =KupiecBackTest(returns = rendimientos_lp$QQQ*M, riskValues = CVar95.garch.norm, alpha = alpha)
garch.t.KT =KupiecBackTest(returns = rendimientos_lp$QQQ*M, riskValues = CVar95.garch.t, alpha = alpha)
garch.ged.KT =KupiecBackTest(returns = rendimientos_lp$QQQ*M, riskValues = CVar95.garch.ged, alpha = alpha)
gjrgarch.norm.KT=KupiecBackTest(returns = rendimientos_lp$QQQ*M, riskValues = CVar95.gjrgarch.norm, alpha = alpha)
gjrgarch.t.KT=KupiecBackTest(returns = rendimientos_lp$QQQ*M, riskValues = CVar95.gjrgarch.t, alpha = alpha)
gjrgarch.ged.KT=KupiecBackTest(returns = rendimientos_lp$QQQ*M, riskValues = CVar95.gjrgarch.ged, alpha = alpha)

```

```

tabla_Kupiec = data.frame(
  Modelo = c("TFS", "EWS95", "EWS98", "garch.norm", "garch.t", "garch.ged", "gjrgarch.norm", "gjrgarch.t", "gjrgarch.ged"),
  Violaciones = c(TFS.KT$Statistic, EWS95.KT$Statistic, EWS98.KT$Statistic, garch.norm.KT$Statistic, garch.t.KT$Statistic, garch.ged.KT$Statistic, gjrgarch.norm.KT$Statistic, gjrgarch.t.KT$Statistic, gjrgarch.ged.KT$Statistic),
  Intervalo = c(TFS.KT$twoSidedCriticalValue, EWS95.KT$twoSidedCriticalValue, EWS98.KT$twoSidedCriticalValue, garch.norm.KT$twoSidedCriticalValue, garch.t.KT$twoSidedCriticalValue, garch.ged.KT$twoSidedCriticalValue, gjrgarch.norm.KT$twoSidedCriticalValue, gjrgarch.t.KT$twoSidedCriticalValue, gjrgarch.ged.KT$twoSidedCriticalValue),
  p_value = c(TFS.KT$pValue, EWS95.KT$pValue, EWS98.KT$pValue, garch.norm.KT$pValue, garch.t.KT$pValue, garch.ged.KT$pValue, gjrgarch.norm.KT$pValue, gjrgarch.t.KT$pValue, gjrgarch.ged.KT$pValue)
)

```

```

CVar95.plot.TFS = plot_ly() %>%
  add_trace(data = rendimientos_lp, x = ~rendimientos_lp$Date, y = ~CVar95.TFS,
    type = "scatter", mode = 'lines', name = "CVaR 95% TFS") %>%
  add_trace(data = rendimientos_lp, x = ~rendimientos_lp$Date, y = ~M*rendimientos_lp$QQQ,
    type = "scatter", mode = 'lines', name = "P&L")
CVar95.plot.EWS95 = plot_ly() %>%

```

```

add_trace(data = rendimientos_lp, x = ~rendimientos_lp$Date, y= ~CVar95.EWS95,
          type = "scatter", mode = 'lines', name = "CVar 95% EWS95") %>%
add_trace(data = rendimientos_lp, x = ~rendimientos_lp$Date, y= ~M*rendimientos_lp$QQQ,
          type = "scatter", mode = 'lines', name = "P&L")
CVar95.plot.EWS98 = plot_ly() %>%
  add_trace(data = rendimientos_lp, x = ~rendimientos_lp$Date, y= ~CVar95.EWS98,
            type = "scatter", mode = 'lines', name = "CVar 95% EWS98") %>%
  add_trace(data = rendimientos_lp, x = ~rendimientos_lp$Date, y= ~M*rendimientos_lp$QQQ,
            type = "scatter", mode = 'lines', name = "P&L")
CVar95.plot.garch.norm = plot_ly() %>%
  add_trace(data = rendimientos_lp, x = ~rendimientos_lp$Date, y= ~CVar95.garch.norm,
            type = "scatter", mode = 'lines', name = "CVar 95% garch.norm") %>%
  add_trace(data = rendimientos_lp, x = ~rendimientos_lp$Date, y= ~M*rendimientos_lp$QQQ,
            type = "scatter", mode = 'lines', name = "P&L")
CVar95.plot.garch.t = plot_ly() %>%
  add_trace(data = rendimientos_lp, x = ~rendimientos_lp$Date, y= ~CVar95.garch.t,
            type = "scatter", mode = 'lines', name = "CVar 95% garch.t") %>%
  add_trace(data = rendimientos_lp, x = ~rendimientos_lp$Date, y= ~M*rendimientos_lp$QQQ,
            type = "scatter", mode = 'lines', name = "P&L")
CVar95.plot.garch.ged = plot_ly() %>%
  add_trace(data = rendimientos_lp, x = ~rendimientos_lp$Date, y= ~CVar95.garch.ged,
            type = "scatter", mode = 'lines', name = "CVar 95% garch.ged") %>%
  add_trace(data = rendimientos_lp, x = ~rendimientos_lp$Date, y= ~M*rendimientos_lp$QQQ,
            type = "scatter", mode = 'lines', name = "P&L")
CVar95.plot.gjrgarch.norm = plot_ly() %>%
  add_trace(data = rendimientos_lp, x = ~rendimientos_lp$Date, y= ~CVar95.gjrgarch.norm,
            type = "scatter", mode = 'lines', name = "CVar 95% gjrgarch.norm") %>%
  add_trace(data = rendimientos_lp, x = ~rendimientos_lp$Date, y= ~M*rendimientos_lp$QQQ,
            type = "scatter", mode = 'lines', name = "P&L")
CVar95.plot.gjrgarch.t = plot_ly() %>%
  add_trace(data = rendimientos_lp, x = ~rendimientos_lp$Date, y= ~CVar95.gjrgarch.t,
            type = "scatter", mode = 'lines', name = "CVar 95% gjrgarch.t") %>%
  add_trace(data = rendimientos_lp, x = ~rendimientos_lp$Date, y= ~M*rendimientos_lp$QQQ,
            type = "scatter", mode = 'lines', name = "P&L")
CVar95.plot.gjrgarch.ged = plot_ly() %>%
  add_trace(data = rendimientos_lp, x = ~rendimientos_lp$Date, y= ~CVar95.gjrgarch.ged,
            type = "scatter", mode = 'lines', name = "CVar 95% gjrgarch.ged") %>%

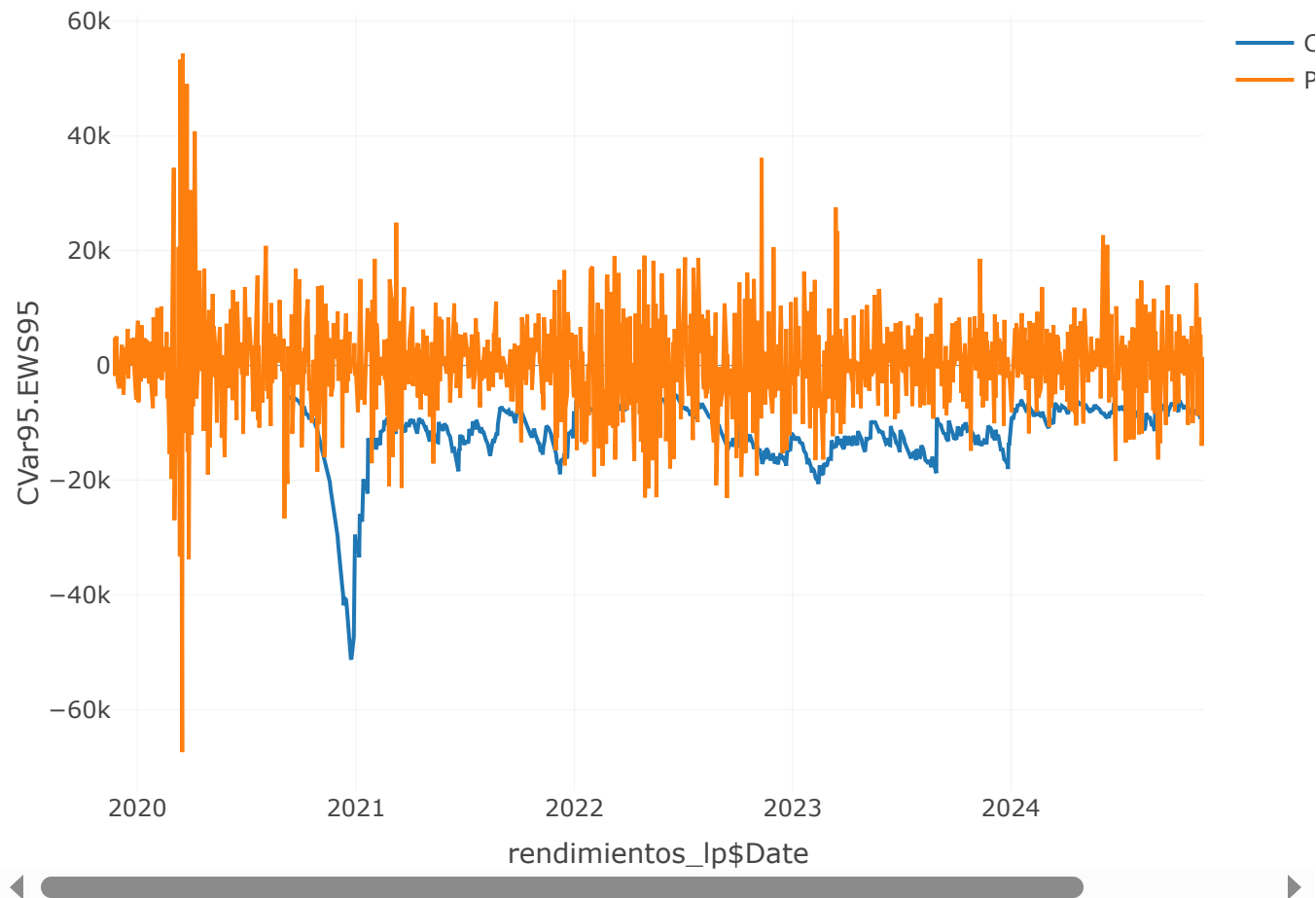
```

```
add_trace(data = rendimientos_lp, x = ~rendimientos_lp$Date, y= ~M*rendimientos_lp$QQQ,  
          type = "scatter", mode = 'lines', name = "P&L")
```

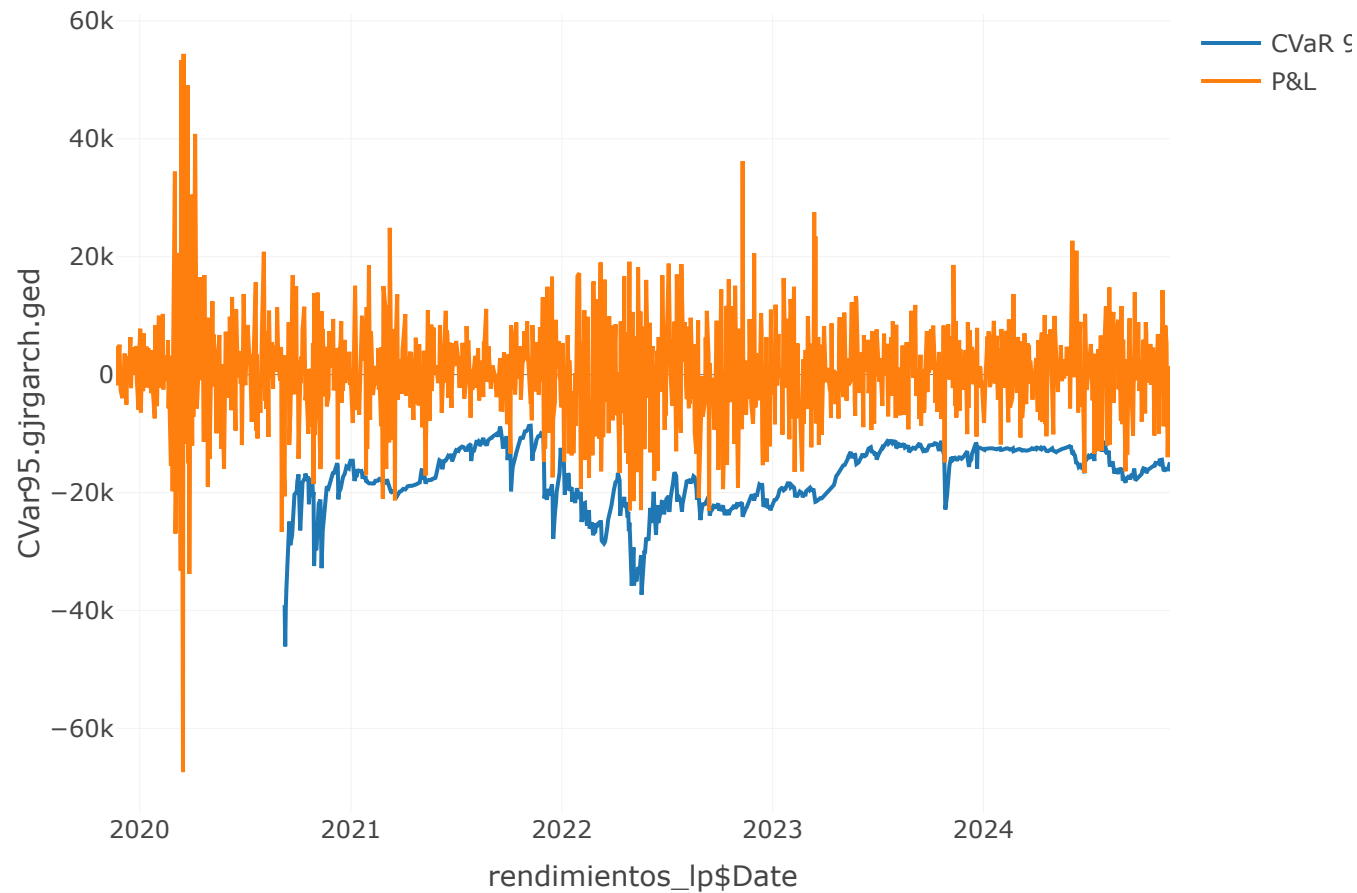
##Comparación Gráfica de Volatilidades y Perdidas Reales

El modelo que más se equivocó fue EWMA con factor de decaimiento 95. Cometió 100 errores. Y haciendo la prueba de Kupiec, debería haber hecho entre 39 y 67

CVaR95.plot.EWS95



El que cometió menos errores fue el GJR-GARCH con errores GED, con 6 violaciones, sin embargo, este sobreestima riesgos, pues debió haber obtenido entre 39 y 67 violaciones.

`CVaR95.plot.gjrgarch.ged`

Curiosamente, ninguno estuvo dentro del intervalo de confianza, y, el más cercano, para mi asombro fue la volatilidad convencional con 33 errores.

`CVaR95.plot.TFS`