Cartera óptima con rendimiento objetivo.

Diseño y análisis.

Dr. Martín Lozano `https://mlozanoqf.github.io/`

03 de enero de 2026, 05:51 p.m.

	Fundamental	Intermedio	Especializado
Finanzas	×	✓	×
Estadística	×	✓	×
R	×	✓	×

1 Introducción.

- Se descargan precios históricos de 10 acciones y se calculan retornos mensuales. Con esos retornos se estiman medias, volatilidades y razón de Sharpe por activo.
- Se construye la frontera eficiente media–varianza y se comparan tres carteras. Finalmente, se evalúa la robustez de riesgo y rendimiento con un block bootstrap de 6 meses (500 réplicas).

2 Diez empresas públicas de Estados Unidos.

Ticker	Nombre de la empresa	Industria
AMD	Advanced Micro Dev.	Computación de alto rendimiento
CNC	Centene Corp.	Servicios de salud
GIS	General Mills, Inc.	Productos de consumo envasados
LMT	Lockheed Martin Corp.	Aeroespacial y defensa
LRCX	Lam Research Corp.	Semiconductores
NEM	Newmont Corp.	Minería de metales preciosos
SNPS	Synopsys, Inc.	Software de diseño
SYF	Synchrony Financial	Servicios financieros
TRMB	Trimble Inc.	Tecnología geoespacial
TTD	The Trade Desk, Inc.	Tecnología publicitaria

3 Paquetes.

```
1 library(tidyquant)
2 library(tidyverse)
3 library(lubridate)
4 library(scales)
5 library(ggrepel)
```

4 Inicialización.

```r
# Empresas y rango temporal

tickers <- c("AMD","CNC","GIS","LMT","LRCX","NEM","SNPS","SYF","TRMB","TTD")
n_months <- 60L
price_start <- ymd("2020-09-01")
price_end <- price_start %m+% months(n_months + 1) - days(1)
returns_start <- price_start %m+% months(1)
returns_end <- returns_start %m+% months(n_months) - months(1)

# Descarga de precios históricos de las acciones

prices <- tq_get(tickers, from = price_start, to = price_end, get = "stock.prices")

# Cálculo de retornos mensuales por activo

monthly_returns <- prices |>
  arrange(symbol, date) |>
  group_by(symbol) |>
  tq_transmute(select = adjusted, mutate_fun = periodReturn,
               period = "monthly", type = "arithmetic",
               col_rename = "monthly_return") |>
  filter(between(date, returns_start, returns_end)) |>
  slice_head(n = n_months) |>
  ungroup()

stopifnot(n_distinct(monthly_returns$symbol) == length(tickers))

# Función: estadísticas básicas por activo

asset_stats <- function(data) {
  data |>
    group_by(symbol) |>
    summarise(ER = mean(monthly_return, na.rm = TRUE),
              SD = sd(monthly_return, na.rm = TRUE),
              SR = ER / SD, .groups = "drop")}

# Función: para rendimiento acumulado

portfolio_series <- function(data, weights, label, start_date) {
  data |>
    filter(symbol %in% names(weights)) |>
    group_by(date) |>
    summarise(portfolio_return = sum(monthly_return * weights[symbol]), .groups = "drop") |>
    arrange(date) |>
    mutate(index_level = cumprod(1 + portfolio_return), symbol = label) |>
    select(date, symbol, index_level) |>
    bind_rows(tibble(date = start_date, symbol = label, index_level = 1))}
```

5 Riesgo-rendimiento: 10 activos individuales.

Valores en unidades decimales.

```
stats <- asset_stats(monthly_returns) |>
  arrange(-SR) |>
  print()
```

```
## # A tibble: 10 x 4
##    symbol         ER     SD        SR
##    <chr>       <dbl>  <dbl>    <dbl>
##  1 SYF     0.0259   0.106   0.244
##  2 SNPS    0.0216   0.0902  0.240
##  3 LRCX    0.0257   0.111   0.231
##  4 AMD     0.0223   0.150   0.149
##  5 TRMB    0.0132   0.0970  0.136
##  6 LMT     0.00709  0.0647  0.110
##  7 NEM     0.0107   0.103   0.103
##  8 TTD     0.0170   0.184   0.0926
##  9 GIS     0.000182 0.0477  0.00382
## 10 CNC    -0.00542  0.102  -0.0532
```

6 Cartera de mínima varianza global.

Sea n el número de activos y R sus rendimientos. Definimos:

$$\Sigma := \mathrm{Var}(R) \in \mathbb{R}^{n\times n}, \qquad \mu := \mathbb{E}[R] \in \mathbb{R}^n, \qquad \mathbf{1} := (1, \dots, 1)^\top \in \mathbb{R}^n.$$

Definimos los escalares:

$$\begin{aligned} a &:= \mathbf{1}^\top \Sigma^{-1} \mathbf{1}, \\ b &:= \mathbf{1}^\top \Sigma^{-1} \mu, \\ c &:= \mu^\top \Sigma^{-1} \mu, \\ d &:= ac - b^2. \end{aligned}$$

El problema es:

$$\min_{w\in\mathbb{R}^n} \; w^\top \Sigma w \quad \text{s.a.} \quad \mathbf{1}^\top w = 1.$$

El Lagrangiano es:

$$\mathcal{L}(w, \lambda) = w^\top \Sigma w - \lambda(\mathbf{1}^\top w - 1).$$

La condición de primer orden (FOC) es:

$$\nabla_w \mathcal{L} = 2\Sigma w - \lambda \mathbf{1} = 0 \quad \Rightarrow \quad w = \frac{\lambda}{2} \Sigma^{-1} \mathbf{1}.$$

Usando la restricción $\mathbf{1}^\top w = 1$:

$$\mathbf{1}^\top w = \frac{\lambda}{2} \mathbf{1}^\top \Sigma^{-1} \mathbf{1} = 1 \quad \Rightarrow \quad \frac{\lambda}{2} = \frac{1}{a}.$$

Por tanto, los pesos de mínima varianza global son:

$$w_{\min} = \frac{\Sigma^{-1}\mathbf{1}}{a}.$$

7 Cartera de mínima varianza con objetivo r^*.

Sea $r^* \in \mathbb{R}$ el rendimiento objetivo. El problema es:

$$\min_{w \in \mathbb{R}^n} \; w^\top \Sigma w \quad \text{s.a.} \quad \mathbf{1}^\top w = 1, \;\; \mu^\top w = r^*.$$

El Lagrangiano es:

$$\mathcal{L}(w, \lambda, \gamma) = w^\top \Sigma w - \lambda(\mathbf{1}^\top w - 1) - \gamma(\mu^\top w - r^*).$$

La FOC es:

$$\nabla_w \mathcal{L} = 2\Sigma w - \lambda \mathbf{1} - \gamma \mu = 0.$$

Definimos los vectores:

$$g := \Sigma^{-1} \frac{c\,\mathbf{1} - b\,\mu}{d}, \qquad h := \Sigma^{-1} \frac{a\,\mu - b\,\mathbf{1}}{d}.$$

Entonces, para un objetivo r^*, los pesos se obtienen como:

$$w(r^*) = g + h\,r^*.$$

Para cualquier vector de pesos w:

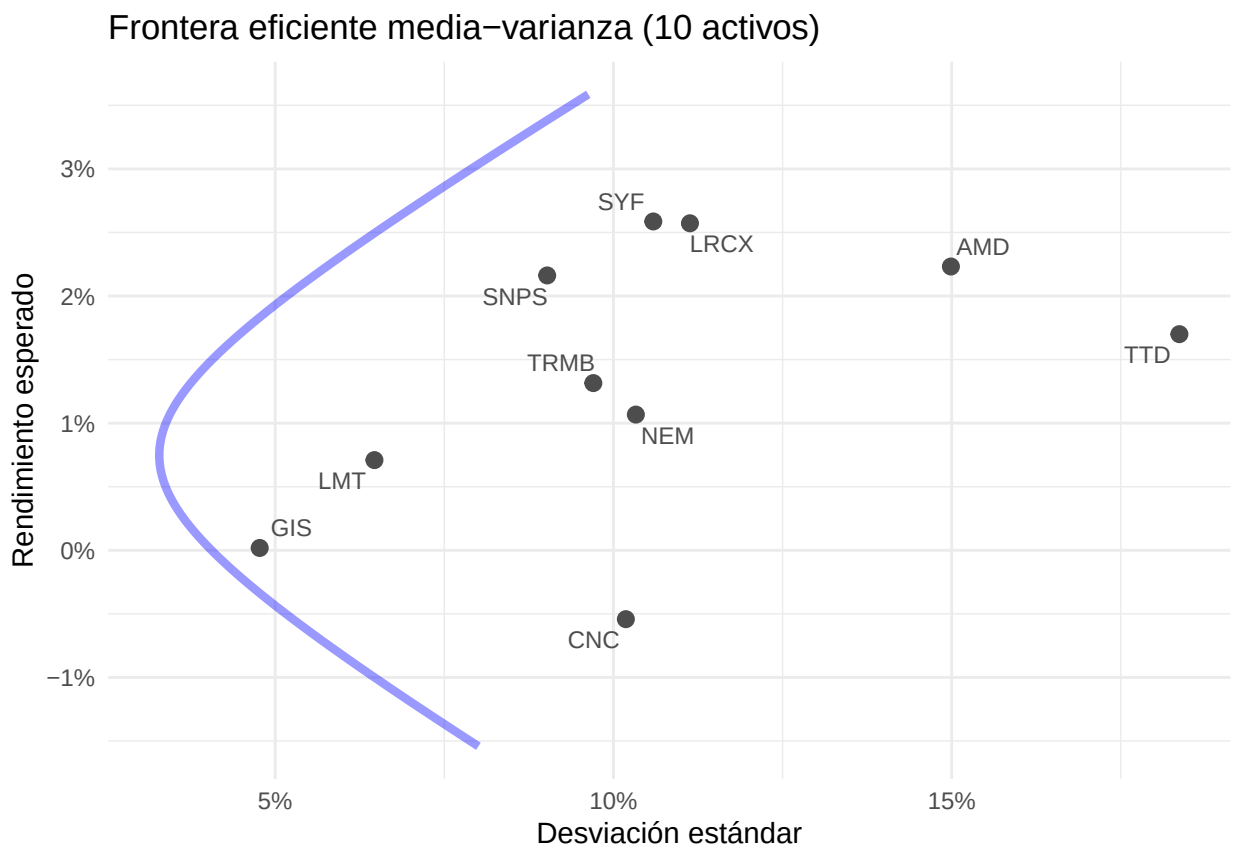
$$\mathbb{E}[R_p] = w^\top \mu, \qquad \sigma_p = \sqrt{w^\top \Sigma w}.$$

8 Del planteamiento matricial al código.

```r
# Matriz de retornos mensuales (filas: meses, columnas: activos)
R <- monthly_returns |>
  select(date, symbol, monthly_return) |>
  pivot_wider(names_from = symbol, values_from = monthly_return) |>
  arrange(date) |>
  select(-date) |>
  drop_na() |>
  as.matrix()

stocks <- colnames(R)
sigma <- var(R)
sd <- diag(sigma)^0.5
E <- colMeans(R)
ones <- rep(1, length(E))

a <- c(t(ones) %*% solve(sigma) %*% ones)
b <- c(t(ones) %*% solve(sigma) %*% E)
c <- c(t(E) %*% solve(sigma) %*% E)
d <- c(a * c - (b^2))

g <- c(solve(sigma) %*% (c * ones - b * E) / d)
h <- c(solve(sigma) %*% (a * E - b * ones) / d)
```

9 Frontera eficiente 10 activos.

```r
ER <- seq(min(E) - 0.01, max(E) + 0.01, length.out = 400)
S <- ER
W <- matrix(0, nrow = length(ER), ncol = length(stocks))

for (i in seq_along(ER)) {
  W[i, ] <- g + h * ER[i]
  ER[i] <- W[i, ] %*% E
  S[i] <- (t(W[i, ]) %*% sigma %*% W[i, ])^0.5}

assets <- tibble(symbol = stocks, sd = sd, ER = E)
frontier <- tibble(S = S, ER = ER)

ggplot() +
    geom_point(data = assets, aes(x = sd, y = ER), color = "grey30", size = 2.5) +
    geom_text_repel(data = assets, aes(x = sd, y = ER, label = symbol),
                    size = 3, color = "grey30") +
    geom_path(data = frontier, aes(x = S, y = ER), color = "blue",
              linewidth = 1.5, alpha = 0.4) +
    labs(title = "Frontera eficiente media-varianza (10 activos)",
      x = "Desviación estándar",
      y = "Rendimiento esperado") +
      scale_x_continuous(labels = scales::percent_format(accuracy = 1)) +
      scale_y_continuous(labels = scales::percent_format(accuracy = 1)) +
    theme_minimal()
```

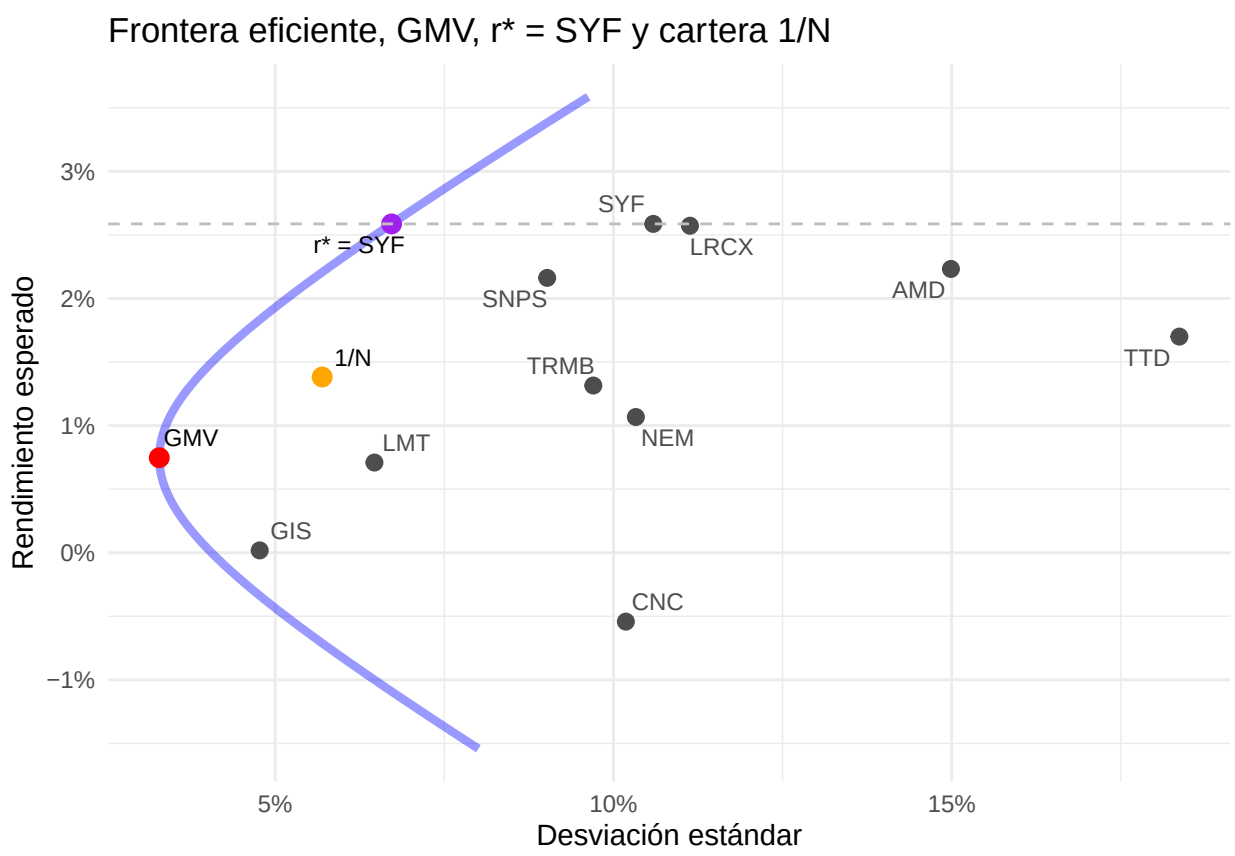


10 Estimación de 3 carteras.

```r
# GMV
W_min <- (solve(sigma) %*% ones) / a
R_min <- c(t(W_min) %*% E)
S_min <- c(t(W_min) %*% sigma %*% W_min)^0.5

# Cartera objetivo: replicar rendimiento esperado de SYF
r_target <- E[stocks == "SYF"]
W_syf <- g + h * r_target
R_syf <- c(t(W_syf) %*% E)
S_syf <- c(t(W_syf) %*% sigma %*% W_syf)^0.5

# Cartera ingenua 1/N
W_1n <- ones / length(E)
R_1n <- c(t(W_1n) %*% E)
S_1n <- c(t(W_1n) %*% sigma %*% W_1n)^0.5
```

11 Frontera eficiente con 3 carteras.

```
assets <- tibble(symbol = stocks, sd = sd, ER = E)
frontier <- tibble(S = S, ER = ER)
ports <- tibble(label = c("GMV", "r* = SYF", "1/N"),
                S = c(S_min, S_syf, S_1n),
                ER = c(R_min, R_syf, R_1n))

ggplot() +
    geom_point(data = assets, aes(x = sd, y = ER), color = "grey30", size = 2.5) +
    geom_text_repel(data = assets, aes(x = sd, y = ER, label = symbol),
                    size = 3, color = "grey30") +
    geom_path(data = frontier, aes(x = S, y = ER), color = "blue",
              linewidth = 1.5, alpha = 0.4) +
    geom_point(data = ports, aes(x = S, y = ER),
               color = c("red", "purple", "orange"), size = 3) +
    geom_text_repel(data = ports, aes(x = S, y = ER, label = label),
                    size = 3, color = "black") +
    geom_hline(yintercept = r_target, linetype = "dashed", color = "grey") +
    labs(title = "Frontera eficiente, GMV, r* = SYF y cartera 1/N",
      x = "Desviación estándar",
      y = "Rendimiento esperado") +
      scale_x_continuous(labels = scales::percent_format(accuracy = 1)) +
      scale_y_continuous(labels = scales::percent_format(accuracy = 1)) +
    theme_minimal()
```



12 Resumen de las carteras.

```r
ER_syf_asset <- E[stocks == "SYF"]
SD_syf_asset <- sd[stocks == "SYF"]

tabla_port <- tibble(cartera = c("1/N", "r* = SYF", "GMV", "SYF"),
                     ER = c(R_1n, R_syf, R_min, ER_syf_asset),
                     SD = c(S_1n, S_syf, S_min, SD_syf_asset)) |>
  mutate(SR = ER / SD) |>
  arrange(desc(SR))

tabla_port
```

```
## # A tibble: 4 x 4
##   cartera       ER     SD    SR
##   <chr>      <dbl>  <dbl> <dbl>
## 1 r* = SYF 0.0259  0.0672 0.385
## 2 SYF      0.0259  0.106  0.244
## 3 1/N      0.0138  0.0569 0.243
## 4 GMV      0.00747 0.0329 0.227
```

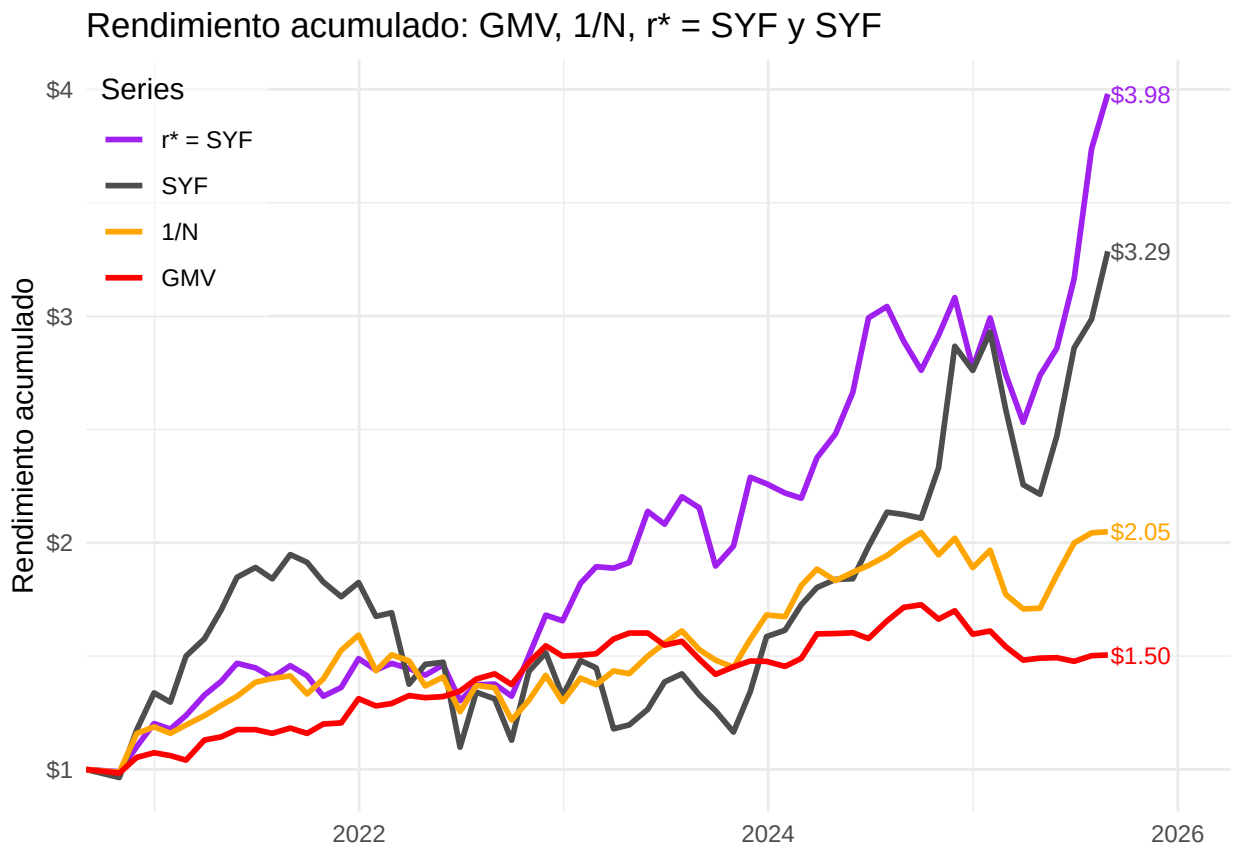
13 Los pesos de las carteras.

```
tabla_pesos <- tibble(symbol = stocks,
                      w_1N = W_1n,
                      w_GMV = c(W_min),
                      w_r_SYF = c(W_syf))

tabla_pesos
```

```
## # A tibble: 10 x 4
##    symbol  w_1N    w_GMV w_r_SYF
##    <chr>  <dbl>    <dbl>   <dbl>
##  1 AMD      0.1 -0.0893  -0.216
##  2 CNC      0.1  0.00510 -0.293
##  3 GIS      0.1  0.539    0.182
##  4 LMT      0.1  0.106    0.248
##  5 LRCX     0.1  0.0416   0.331
##  6 NEM      0.1  0.0419   0.167
##  7 SNPS     0.1  0.240    0.500
##  8 SYF      0.1  0.0187   0.388
##  9 TRMB     0.1  0.0490  -0.360
## 10 TTD      0.1  0.0469   0.0543
```

14 Rendimiento acumulado de las carteras.

```r
names(W_min) <- stocks
names(W_1n) <- stocks
names(W_syf) <- stocks

# Pesos de SYF al 100%
W_syf_asset <- rep(0, length(stocks))
names(W_syf_asset) <- stocks
W_syf_asset[stocks == "SYF"] <- 1

weights_specs <- list("GMV" = W_min,
                      "1/N" = W_1n,
                      "r* = SYF" = W_syf,
                      "SYF" = W_syf_asset)

portfolio_cum <- purrr::imap_dfr(weights_specs,
                                 ~ portfolio_series(monthly_returns, .x, .y, price_start))

last_values_port <- portfolio_cum |>
  group_by(symbol) |>
  slice_max(order_by = date, n = 1, with_ties = FALSE) |>
  select(symbol, final_value = index_level) |>
  arrange(desc(final_value))

series_levels <- last_values_port$symbol

palette_base <- c("GMV" = "red",
                  "1/N" = "orange",
                  "r* = SYF" = "purple",
                  "SYF" = "grey30")

portfolio_colors <- palette_base[series_levels]

portfolio_cum <- portfolio_cum |>
  mutate(symbol = factor(symbol, levels = series_levels))

label_data_port <- portfolio_cum |>
  group_by(symbol) |>
  slice_max(order_by = date, n = 1, with_ties = FALSE) |>
  mutate(label = paste0("$", formatC(index_level, format = "f", digits = 2))) |>
  ungroup()

ggplot(portfolio_cum, aes(date, index_level, color = symbol)) +
  geom_line(linewidth = 1) +
  geom_text(data = label_data_port, aes(label = label),
            hjust = -0.05, vjust = 0.5, size = 3.2, show.legend = FALSE) +
  scale_color_manual(values = portfolio_colors, breaks = series_levels,
                     limits = series_levels, drop = FALSE) +
  scale_x_date(expand = expansion(mult = c(0, 0.12))) +
  scale_y_continuous(labels = scales::dollar_format(prefix = "$")) +
  labs(title = "Rendimiento acumulado: GMV, 1/N, r* = SYF y SYF",
       y = "Rendimiento acumulado", x = NULL, color = "Series") +
    theme_minimal() +
    theme(legend.position = c(0, 1), legend.justification = c(0, 1),
          legend.background = element_rect(fill = alpha("white", 0.6), color = NA))
```


Rendimiento acumulado: GMV, 1/N, r* = SYF y SYF
Series
r* = SYF
SYF
1/N
GMV
Rendimiento acumulado
$4
$3
$2
$1
2022
2024
2026
$3.98
$3.29
$2.05
$1.50

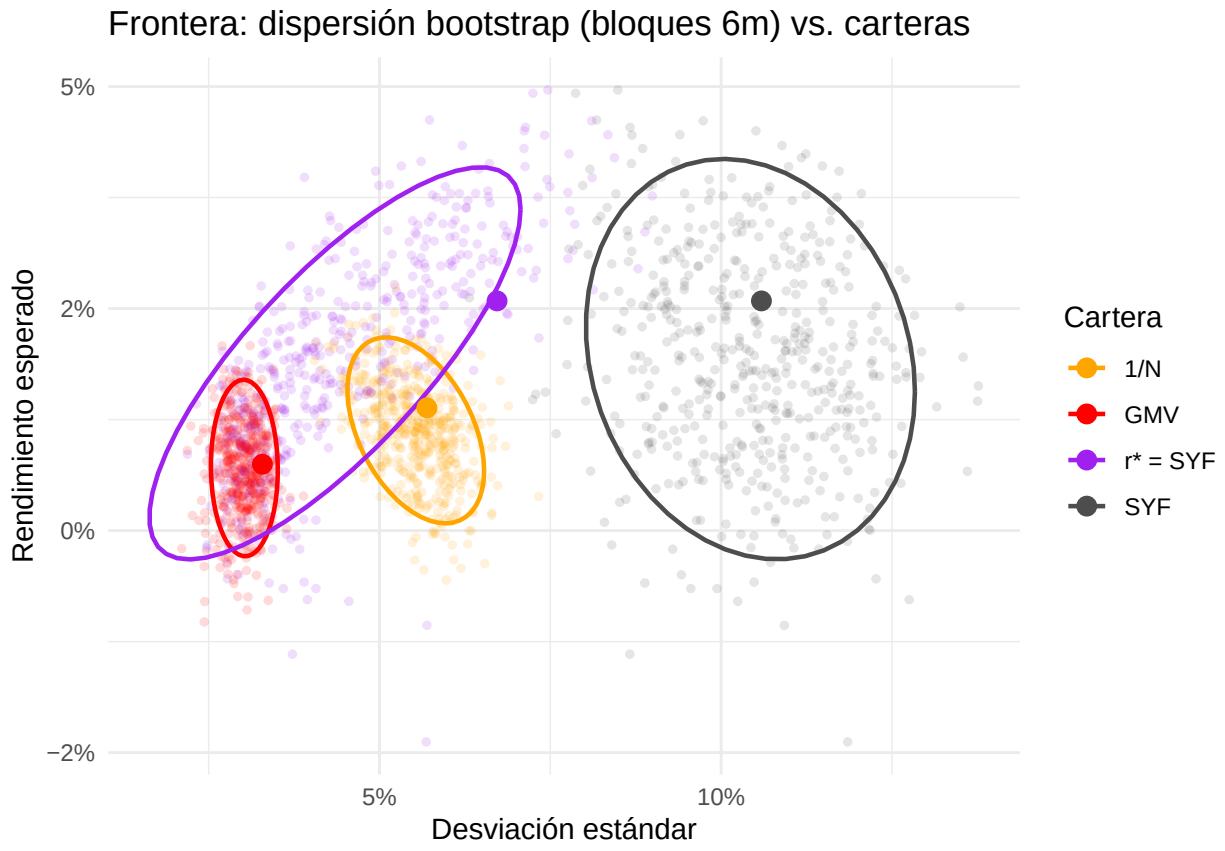
15 Escenarios de media–varianza por block bootstrap.

Block bootstrap con 500 réplicas. De los 60 meses históricos se toman bloques contiguos de 6 meses y se remezclan con reemplazo hasta completar 60 meses por réplica. En cada réplica se recalculan las cuatro carteras, obteniendo 500 puntos por cartera. Así se preserva la estructura temporal dentro de cada bloque, mitigando la mezcla de meses sueltos; solo cambia el orden de los bloques.

```
set.seed(123)
block_len <- 6
B <- 500
n <- nrow(R)

# Lista de bloques posibles (matrices de block_len filas)
blocks <- map(1:(n - block_len + 1),
              ~ R[.x:(.x + block_len - 1), , drop = FALSE])

# Función para una réplica bootstrap en bloques
sample_block_matrix <- function() {
  out <- NULL
  filled <- 0
  while (filled < n) {blk <- sample(blocks, 1)[[1]]
    out <- if (is.null(out)) blk else rbind(out, blk)
    filled <- nrow(out)}
  out[seq_len(n), , drop = FALSE]}

boot_stats <- map_dfr(seq_len(B), function(bi) {
  Rb <- sample_block_matrix()
  Eb <- colMeans(Rb)
  sigb <- var(Rb)
  ones_b <- rep(1, length(Eb))

a_b <- c(t(ones_b) %*% solve(sigb) %*% ones_b)
b_b <- c(t(ones_b) %*% solve(sigb) %*% Eb)
c_b <- c(t(Eb) %*% solve(sigb) %*% Eb)
d_b <- c(a_b * c_b - (b_b^2))
g_b <- c(solve(sigb) %*% (c_b * ones_b - b_b * Eb) / d_b)
h_b <- c(solve(sigb) %*% (a_b * Eb - b_b * ones_b) / d_b)

r_target_b <- Eb[stocks == "SYF"]
W_min_b <- (solve(sigb) %*% ones_b) / a_b
W_1n_b <- ones_b / length(Eb)
W_syf_b <- g_b + h_b * r_target_b
W_syf_asset_b <- rep(0, length(Eb)); W_syf_asset_b[stocks == "SYF"] <- 1

tibble(cartera = c("GMV", "1/N", "r* = SYF", "SYF"),
       ER = c(t(W_min_b) %*% Eb, t(W_1n_b) %*% Eb,
              t(W_syf_b) %*% Eb, t(W_syf_asset_b) %*% Eb),
       SD = c(t(W_min_b) %*% sigb %*% W_min_b,
              t(W_1n_b) %*% sigb %*% W_1n_b,
              t(W_syf_b) %*% sigb %*% W_syf_b,
              t(W_syf_asset_b) %*% sigb %*% W_syf_asset_b)^0.5)})
```

16 Gráfico del block bootstrap.

```r
# Puntos originales de referencia
ref_ports <- tibble(cartera = c("GMV", "1/N", "r* = SYF", "SYF"),
                    ER = c(R_min, R_1n, R_syf, ER_syf_asset),
                    SD = c(S_min, S_1n, S_syf, SD_syf_asset))
palette_boot <- c("GMV" = "red", "1/N" = "orange",
                  "r* = SYF" = "purple", "SYF" = "grey30")

ggplot(boot_stats, aes(SD, ER, color = cartera)) +
  geom_point(alpha = 0.15, size = 1) +
  stat_ellipse(level = 0.9, type = "t", linewidth = 0.8) +
  geom_point(data = ref_ports, aes(SD, ER, color = cartera), size = 3) +
  scale_color_manual(values = palette_boot) +
  scale_x_continuous(labels = percent_format(accuracy = 1)) +
  scale_y_continuous(labels = percent_format(accuracy = 1)) +
  labs(title = "Frontera: dispersión bootstrap (bloques 6m) vs. carteras",
       x = "Desviación estándar", y = "Rendimiento esperado", color = "Cartera") +
  theme_minimal()
```



El gráfico muestra la nube de esos puntos y una elipse 90% por cartera, junto a los puntos históricos, para apreciar cuánta variabilidad pueden tener media y riesgo bajo cambios plausibles del régimen. Elipses pequeñas sugieren mayor robustez, elipses grandes indican dependencia del periodo.

17 Conclusión.

- Los rendimientos acumulados permiten comparar desempeño histórico más allá de un solo punto riesgo–retorno.
- El block bootstrap muestra qué tan estables son las carteras ante cambios plausibles del periodo.