# Carteras.

xxxx.

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

21 de noviembre de 2025, 01:04 a.m.

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

# 1 Introducción.

- Estimar la volatilidad no condicional de los rendimientos del índice S&P 500, utilizando datos históricos de precios diarios.
- Rendimientos como cambios porcentuales y logaritmos naturales.
- Base para futuros modelos financieros: Black–Scholes, Markowitz, VaR, Monte Carlo, Random Walk, Geometric Brownian Motion, Vasicek, riesgo de crédito de Merton, CAPM, APT.

## 2 Paquetes.

| Ticker | Nombre de la empresa | Industria |
|---|---|---|
| AMD | Advanced Micro Devices, Inc. | Semiconductores / Computación de alto rendimiento |
| CNC | Centene Corporation | Servicios de salud / Seguros de salud administrados |
| GIS | General Mills, Inc. | Alimentación / Productos de consumo envasados |
| LMT | Lockheed Martin Corporation | Aeroespacial y defensa |
| LRCX | Lam Research Corporation | Equipos para fabricación de semiconductores |
| NEM | Newmont Corporation | Minería de metales preciosos (oro, cobre, etc.) |
| SNPS | Synopsys, Inc. | Software de automatización de diseño electrónico (EDA) |
| SYF | Synchrony Financial | Servicios financieros al consumidor / Crédito al consumo |
| TRMB | Trimble Inc. | Tecnología geoespacial, construcción y transporte |
| TTD | The Trade Desk, Inc. | Tecnología publicitaria / Plataforma de compra de anuncios |

```r
library(tidyquant)
  library(tidyverse)
  library(lubridate)

  tickers       <- c("AMD","CNC","GIS","LMT","LRCX","NEM","SNPS","SYF","TRMB","TTD")
  n_months      <- 60L
  price_start   <- ymd("2020-09-01")                  # un mes extra para generar los retornos
  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)

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

```

```r
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)) %>%
  mutate(obs = row_number()) %>%
  filter(obs <= n_months) %>%
  select(-obs) %>%
  ungroup()

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

stats <- monthly_returns %>%
  group_by(symbol) %>%
  summarise(
    expected_return = mean(monthly_return),
    sd_return       = sd(monthly_return),
    sharpe_ratio    = expected_return / sd_return,
    .groups = "drop"
  )

returns_wide <- monthly_returns %>%
  select(date, symbol, monthly_return) %>%
  pivot_wider(names_from = symbol, values_from = monthly_return) %>%
  arrange(date)

corr_mat <- returns_wide %>%
  select(-date) %>%
  cor(use = "pairwise.complete.obs")

lowest_pairs <- corr_mat %>%
  as_tibble(rownames = "symbol_i") %>%
  pivot_longer(-symbol_i, names_to = "symbol_j", values_to = "corr") %>%
  filter(symbol_i < symbol_j) %>%
  arrange(corr) %>%
  slice_head(n = 10)

print(stats)
```

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

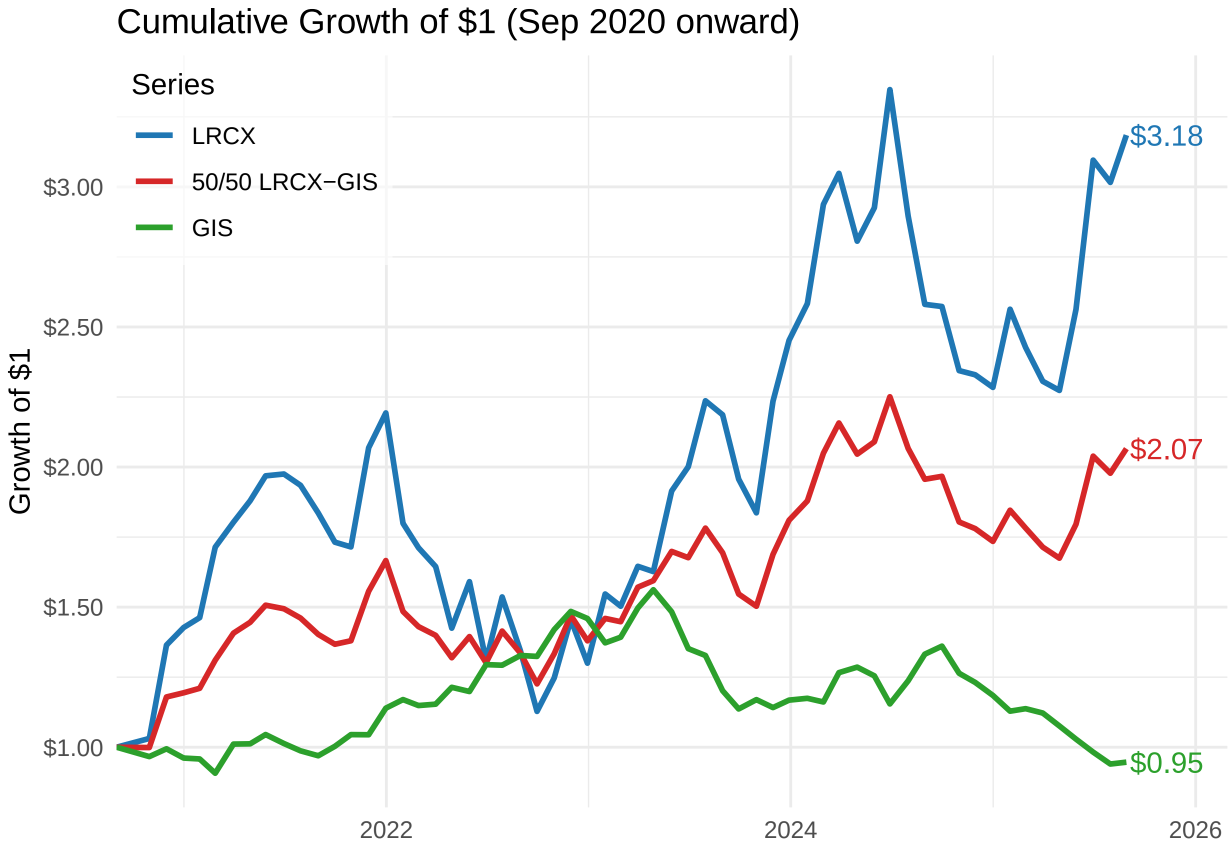
```
corr_mat_fmt <- corr_mat %>%
  round(2) %>%                         # 2 decimales bastan para correlaciones
  format(trim = TRUE, nsmall = 2) %>%  # remueve espacios extra y alinea
  as.matrix()

print(corr_mat_fmt, quote = FALSE, right = TRUE)
```

```
##        AMD   CNC   GIS   LMT  LRCX   NEM  SNPS   SYF  TRMB   TTD
## AMD   1.00 -0.10 -0.16 -0.12  0.71 -0.06  0.70  0.35  0.50  0.51
## CNC  -0.10  1.00  0.37  0.37  0.14  0.20 -0.16  0.12  0.01 -0.18
## GIS  -0.16  0.37  1.00  0.46 -0.20  0.25 -0.27 -0.15 -0.20 -0.20
## LMT  -0.12  0.37  0.46  1.00  0.00  0.32 -0.22  0.20  0.01 -0.10
## LRCX  0.71  0.14 -0.20  0.00  1.00  0.04  0.56  0.59  0.61  0.38
## NEM  -0.06  0.20  0.25  0.32  0.04  1.00 -0.04 -0.08 -0.05 -0.26
## SNPS  0.70 -0.16 -0.27 -0.22  0.56 -0.04  1.00  0.33  0.48  0.31
## SYF   0.35  0.12 -0.15  0.20  0.59 -0.08  0.33  1.00  0.67  0.29
## TRMB  0.50  0.01 -0.20  0.01  0.61 -0.05  0.48  0.67  1.00  0.44
## TTD   0.51 -0.18 -0.20 -0.10  0.38 -0.26  0.31  0.29  0.44  1.00
```

```
print(lowest_pairs)
```

```
## # A tibble: 10 x 3
##    symbol_i symbol_j   corr
##    <chr>    <chr>     <dbl>
##  1 GIS      SNPS     -0.266
##  2 NEM      TTD      -0.263
##  3 LMT      SNPS     -0.217
##  4 GIS      TTD      -0.201
##  5 GIS      LRCX     -0.201
##  6 GIS      TRMB     -0.198
##  7 CNC      TTD      -0.180
##  8 AMD      GIS      -0.162
##  9 CNC      SNPS     -0.156
## 10 GIS      SYF      -0.148
```

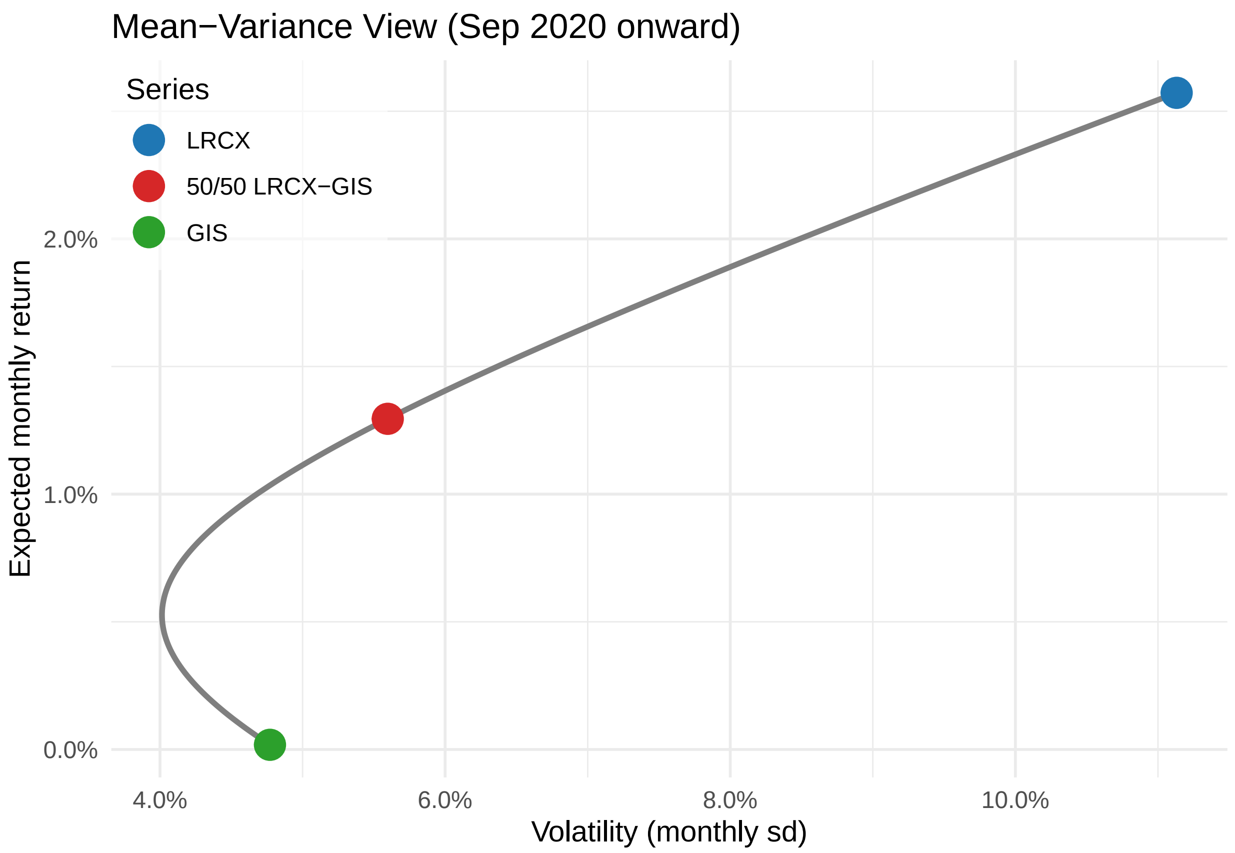
```r
library(scales)

focus_symbols <- c("LRCX", "GIS")
base_rows <- tibble(
  date = price_start,
  symbol = focus_symbols,
  monthly_return = NA_real_,
  index_level = 1
)

cum_returns <- monthly_returns %>%
  filter(symbol %in% focus_symbols) %>%
  arrange(symbol, date) %>%
  group_by(symbol) %>%
  mutate(index_level = cumprod(1 + monthly_return)) %>%
  ungroup() %>%
  bind_rows(base_rows) %>%
  arrange(symbol, date) %>%
  select(date, symbol, index_level)

portfolio <- cum_returns %>%
  pivot_wider(names_from = symbol, values_from = index_level) %>%
  mutate(`50/50 LRCX-GIS` = 0.5 * LRCX + 0.5 * GIS) %>%
  select(date, `50/50 LRCX-GIS`) %>%
  pivot_longer(-date, names_to = "symbol", values_to = "index_level")

plot_data <- bind_rows(cum_returns, portfolio)

last_values <- plot_data %>%
  arrange(date) %>%
  group_by(symbol) %>%
  summarise(final_value = last(index_level), .groups = "drop") %>%
  arrange(desc(final_value))

color_values <- c("#1f77b4", "#d62728", "#2ca02c")
names(color_values) <- last_values$symbol

plot_data <- plot_data %>%
  mutate(symbol = factor(symbol, levels = last_values$symbol))

label_data <- plot_data %>%
  group_by(symbol) %>%
  filter(date == max(date)) %>%
  ungroup() %>%
  mutate(label = paste0("$", formatC(index_level, format = "f", digits = 2)))

ggplot(plot_data, aes(x = date, y = index_level, color = symbol)) +
  geom_line(size = 1) +
  geom_text(
    data = label_data,
    aes(label = label),
    hjust = -0.05,
    vjust = 0.5,
    show.legend = FALSE
  ) +
  scale_color_manual(values = color_values, limits = last_values$symbol) +
  scale_x_date(expand = expansion(mult = c(0, 0.1))) +
  scale_y_continuous(labels = dollar_format(prefix = "$")) +
  labs(
    title = "Cumulative Growth of $1 (Sep 2020 onward)",
    y = "Growth of $1",
    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)
  )
```



```r
weights <- c(LRCX = 0.5, GIS = 0.5)

stats_lrcx_gis <- monthly_returns %>%
  filter(symbol %in% names(weights)) %>%
  group_by(symbol) %>%
  summarise(
    expected_return = mean(monthly_return),
    sd_return       = sd(monthly_return),
    sharpe_ratio    = expected_return / sd_return,
    .groups = "drop"
  )

portfolio_stats <- monthly_returns %>%
  filter(symbol %in% names(weights)) %>%
  mutate(weight = weights[symbol]) %>%
  group_by(date) %>%
  summarise(portfolio_return = sum(monthly_return * weight), .groups = "drop") %>%
  summarise(
    symbol          = "50/50 LRCX-GIS",
    expected_return = mean(portfolio_return),
    sd_return       = sd(portfolio_return),
    sharpe_ratio    = expected_return / sd_return
  )

final_table <- bind_rows(stats_lrcx_gis, portfolio_stats) %>%
  mutate(symbol = factor(symbol, levels = last_values$symbol)) %>%
  arrange(symbol)

print(final_table)
```

```
## # A tibble: 3 x 4
##   symbol         expected_return sd_return sharpe_ratio
##   <fct>                    <dbl>     <dbl>        <dbl>
## 1 LRCX                  0.0257      0.111       0.231
## 2 50/50 LRCX-GIS        0.0130      0.0560      0.231
## 3 GIS                   0.000182    0.0477      0.00382
```

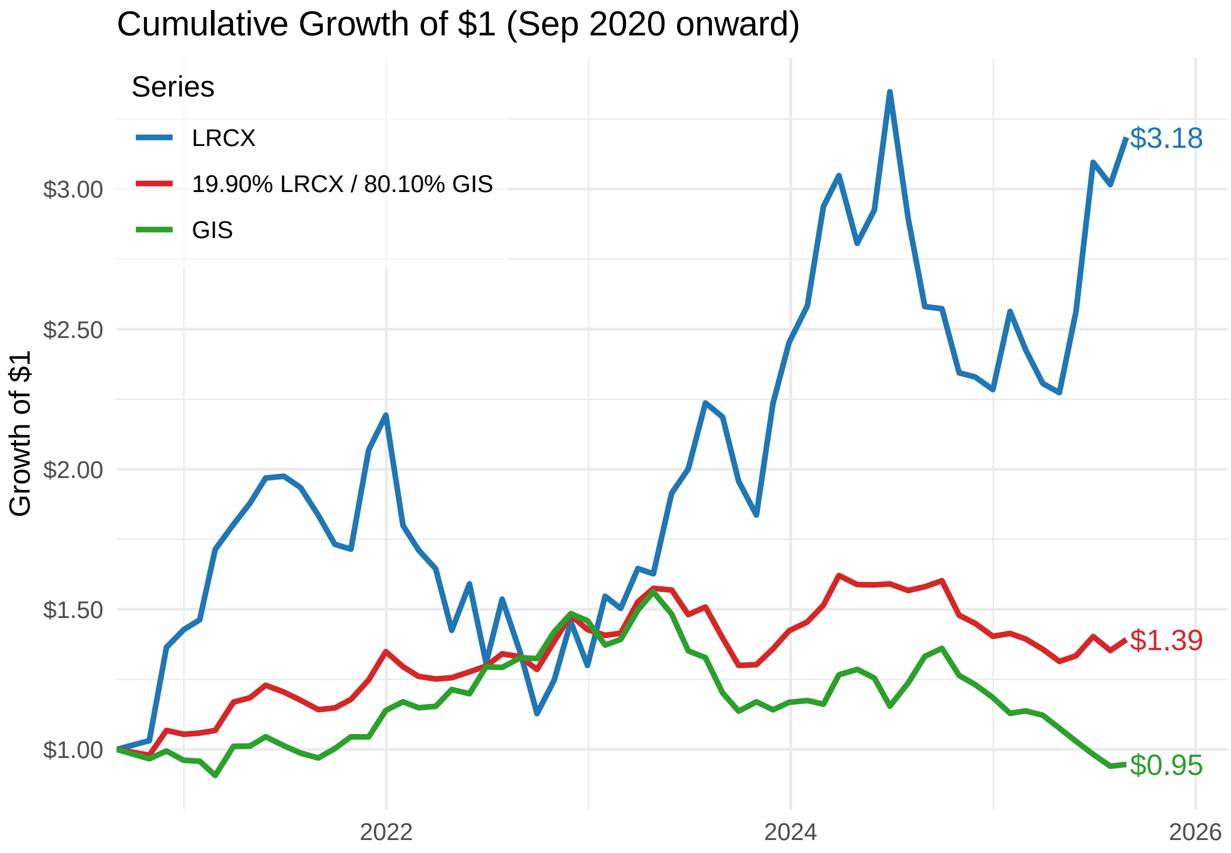
```r
  returns_matrix <- monthly_returns %>%
    filter(symbol %in% names(weights)) %>%
    select(date, symbol, monthly_return) %>%
    pivot_wider(names_from = symbol, values_from = monthly_return) %>%
    arrange(date) %>%
    select(all_of(names(weights)))

  cov_mat <- cov(returns_matrix, use = "pairwise.complete.obs")

  mu_vec <- stats_lrcx_gis %>%
    select(symbol, expected_return) %>%
    deframe()
  sd_vec <- stats_lrcx_gis %>%
    select(symbol, sd_return) %>%
    deframe()

 frontier_data <- tibble(weight_lrcx = seq(0, 1, length.out = 200)) %>%
    mutate(
      weight_gis      = 1 - weight_lrcx,
      expected_return = weight_lrcx * mu_vec["LRCX"] + weight_gis * mu_vec["GIS"],
      variance = weight_lrcx^2 * sd_vec["LRCX"]^2 +
                 weight_gis^2  * sd_vec["GIS"]^2  +
                 2 * weight_lrcx * weight_gis * cov_mat["LRCX", "GIS"],
      sd_return = sqrt(variance)
    )
  # sin arrange() para preservar el orden de los pesos

    portfolio_label <- "19.90% LRCX / 80.10% GIS"
  custom_weights  <- c(LRCX = 0.1990055, GIS = 0.8009945)

 series_order <- c("LRCX", "GIS", "50/50 LRCX-GIS", portfolio_label)
mean_var_points <- final_table %>%
  filter(symbol %in% series_order) %>%           # deja solo los dos activos + dos carteras
  mutate(symbol = factor(symbol, levels = series_order)) %>%
  arrange(symbol)


ggplot() +
  geom_path(
    data = frontier_data,
    aes(x = sd_return, y = expected_return),
    color = "grey50",
    linewidth = 1
  ) +
  geom_point(
    data = mean_var_points,
    aes(x = sd_return, y = expected_return, color = symbol),
    size = 5
  ) +
  scale_color_manual(values = color_values, limits = last_values$symbol) +
  scale_x_continuous(labels = percent_format(accuracy = 0.1)) +
  scale_y_continuous(labels = percent_format(accuracy = 0.1)) +
  labs(
    title = "Mean-Variance View (Sep 2020 onward)",
    x = "Volatility (monthly sd)",
    y = "Expected monthly return",
    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)
  )
```



```r
library(quadprog)

  assets <- names(weights)
  cov_two_assets <- cov_mat[assets, assets]

  Dmat <- 2 * cov_two_assets
  dvec <- rep(0, length(assets))
  Amat <- matrix(1, nrow = length(assets), ncol = 1)  # 1'w = 1
  bvec <- 1

  qp <- solve.QP(Dmat, dvec, Amat, bvec, meq = 1)
  weights_opt <- setNames(qp$solution, assets)

  print(weights_opt)
```

```
##      LRCX       GIS
## 0.1990059 0.8009941
```

```r
  library(scales)

  focus_symbols <- c("LRCX", "GIS")
  custom_weights <- c(LRCX = 0.1990055, GIS = 0.8009945)
  portfolio_label <- "19.90% LRCX / 80.10% GIS"

  base_rows <- tibble(
    date = price_start,
    symbol = focus_symbols,
    monthly_return = NA_real_,
    index_level = 1
  )

  cum_returns <- monthly_returns %>%
    filter(symbol %in% focus_symbols) %>%
    arrange(symbol, date) %>%
    group_by(symbol) %>%
    mutate(index_level = cumprod(1 + monthly_return)) %>%
    ungroup() %>%
    bind_rows(base_rows) %>%
    arrange(symbol, date) %>%
    select(date, symbol, index_level)

portfolio <- cum_returns %>%
    pivot_wider(names_from = symbol, values_from = index_level) %>%
    mutate("{portfolio_label}" := custom_weights["LRCX"] * LRCX +
                                  custom_weights["GIS"]  * GIS) %>%
    select(date, all_of(portfolio_label)) %>%
    pivot_longer(-date, names_to = "symbol", values_to = "index_level")

  plot_data <- bind_rows(cum_returns, portfolio)

  last_values <- plot_data %>%
    arrange(date) %>%
    group_by(symbol) %>%
    summarise(final_value = last(index_level), .groups = "drop") %>%
    arrange(desc(final_value))

  color_values <- c("#1f77b4", "#d62728", "#2ca02c")
  names(color_values) <- last_values$symbol

  plot_data <- plot_data %>%
    mutate(symbol = factor(symbol, levels = last_values$symbol))

  label_data <- plot_data %>%
    group_by(symbol) %>%
    filter(date == max(date)) %>%
    ungroup() %>%
    mutate(label = paste0("$", formatC(index_level, format = "f", digits = 2)))

  ggplot(plot_data, aes(x = date, y = index_level, color = symbol)) +
    geom_line(size = 1) +
    geom_text(
      data = label_data,
      aes(label = label),
      hjust = -0.05,
      vjust = 0.5,
      show.legend = FALSE
    ) +
    scale_color_manual(values = color_values, limits = last_values$symbol) +
    scale_x_date(expand = expansion(mult = c(0, 0.1))) +
    scale_y_continuous(labels = dollar_format(prefix = "$")) +
    labs(
      title = "Cumulative Growth of $1 (Sep 2020 onward)",
      y = "Growth of $1",
      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)
    )
```

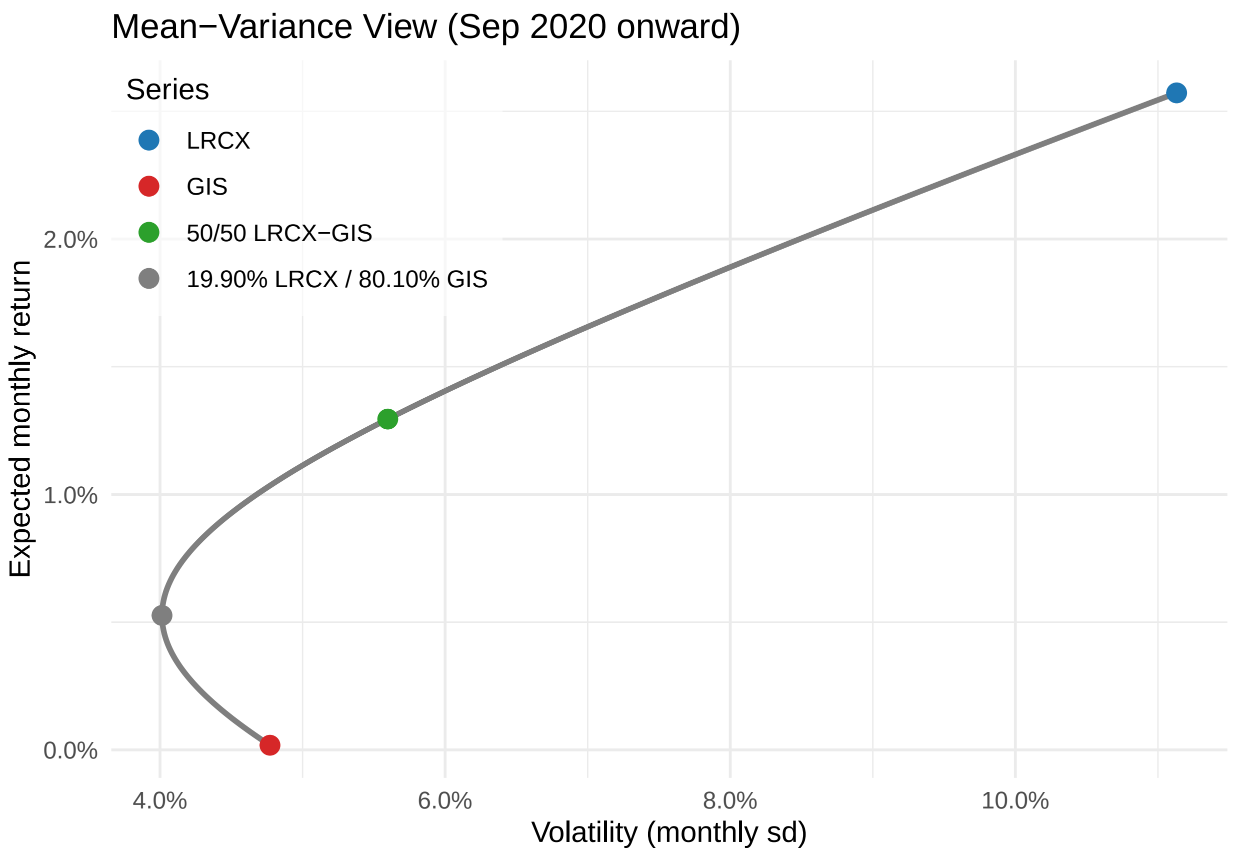


```
  custom_portfolio_stats <- monthly_returns %>%
    filter(symbol %in% names(custom_weights)) %>%
    mutate(weight = custom_weights[symbol]) %>%
    group_by(date) %>%
    summarise(portfolio_return = sum(monthly_return * weight), .groups = "drop") %>%
    summarise(
      symbol          = portfolio_label,
      expected_return = mean(portfolio_return),
      sd_return       = sd(portfolio_return),
      sharpe_ratio    = expected_return / sd_return
    )

table_levels <- c("LRCX", "GIS", "50/50 LRCX-GIS", portfolio_label)

  final_table <- final_table %>%
    mutate(symbol = as.character(symbol)) %>%
    bind_rows(custom_portfolio_stats) %>%
    mutate(symbol = factor(symbol, levels = table_levels)) %>%
```

```r
  arrange(symbol)

print(final_table)
```

```
## # A tibble: 4 x 4
##   symbol                       expected_return sd_return sharpe_ratio
##   <fct>                                  <dbl>     <dbl>        <dbl>
## 1 LRCX                                0.0257      0.111       0.231
## 2 GIS                                 0.000182    0.0477      0.00382
## 3 50/50 LRCX-GIS                      0.0130      0.0560      0.231
## 4 19.90% LRCX / 80.10% GIS            0.00526     0.0401      0.131
```

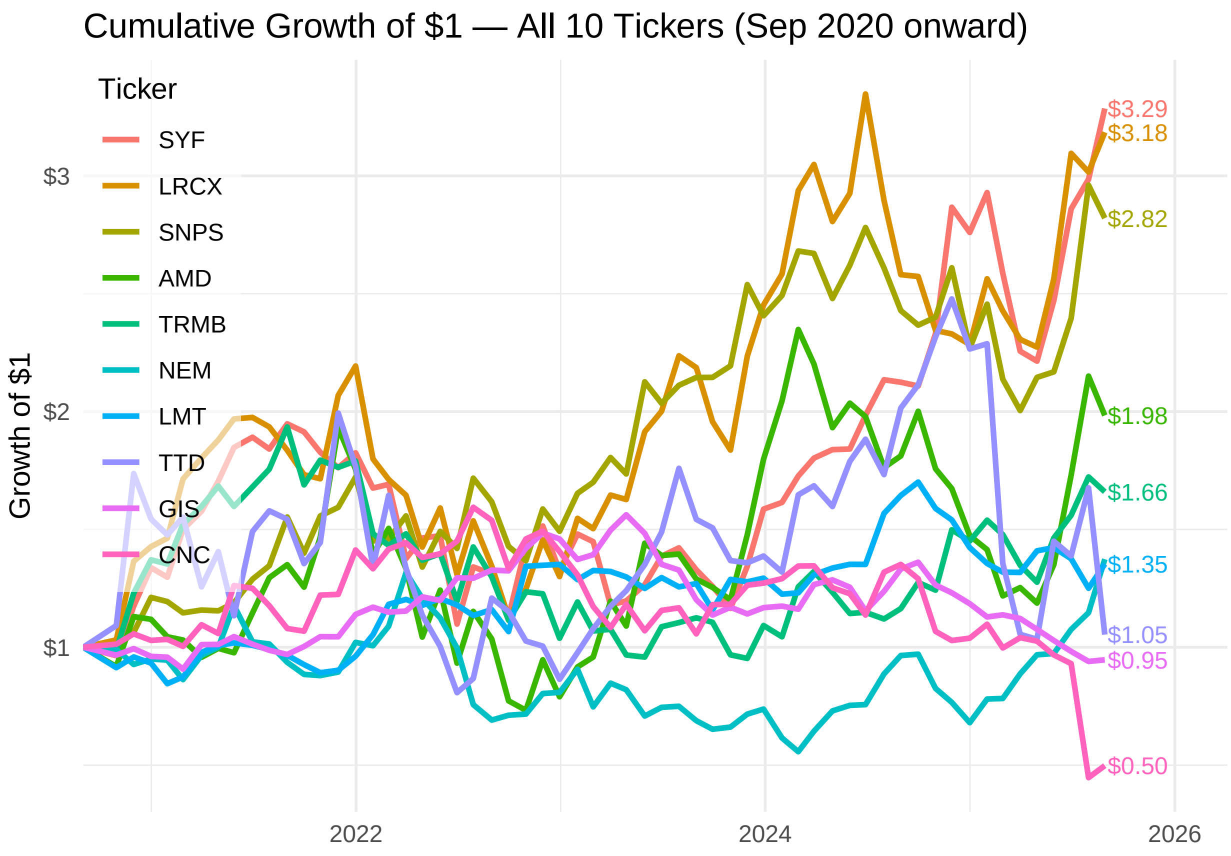
```r
table_levels <- c("LRCX", "GIS", "50/50 LRCX-GIS", portfolio_label)

mean_var_points <- final_table %>%
  mutate(symbol = factor(symbol, levels = table_levels))

color_values_mv <- c(
  "LRCX"                = "#1f77b4",
  "GIS"                 = "#d62728",
  "50/50 LRCX-GIS"      = "#2ca02c",
  portfolio_label       = "#ff7f0e"
)

ggplot() +
  geom_path(
    data = frontier_data,
    aes(x = sd_return, y = expected_return),
    color = "grey50",
    linewidth = 1
  ) +
  geom_point(
    data = mean_var_points,
    aes(x = sd_return, y = expected_return, color = symbol),
    size = 3
  ) +
  scale_color_manual(values = color_values_mv, limits = table_levels) +
  scale_x_continuous(labels = percent_format(accuracy = 0.1)) +
  scale_y_continuous(labels = percent_format(accuracy = 0.1)) +
  labs(
    title = "Mean-Variance View (Sep 2020 onward)",
    x = "Volatility (monthly sd)",
    y = "Expected monthly return",
    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)
  )
```



```r
library(scales)

focus_symbols <- c("LRCX", "GIS")
ci_mult <- 1.96

base_rows <- tibble(
  date = price_start,
  symbol = focus_symbols,
  monthly_return = NA_real_,
  index_level = 1
)

cum_returns <- monthly_returns %>%
  filter(symbol %in% focus_symbols) %>%
  arrange(symbol, date) %>%
  group_by(symbol) %>%
  mutate(index_level = cumprod(1 + monthly_return)) %>%
  ungroup() %>%
  bind_rows(base_rows) %>%
  arrange(symbol, date) %>%
  select(date, symbol, index_level)

final_values <- cum_returns %>%
  group_by(symbol) %>%
  summarise(final_value = last(index_level), .groups = "drop")

stats_assets <- monthly_returns %>%
  filter(symbol %in% focus_symbols) %>%
  group_by(symbol) %>%
  summarise(sd_return = sd(monthly_return), .groups = "drop")

portfolios <- list(
  "50/50 LRCX-GIS" = c(LRCX = 0.5, GIS = 0.5)
)
portfolios[[portfolio_label]] <- custom_weights

stats_portfolios <- bind_rows(lapply(names(portfolios), function(label) {
  w <- portfolios[[label]]

  pr <- monthly_returns %>%
    filter(symbol %in% names(w)) %>%
    mutate(weight = w[symbol]) %>%
    group_by(date) %>%
    summarise(portfolio_return = sum(monthly_return * weight), .groups = "drop") %>%
    arrange(date)

  final_value <- pr %>%
    mutate(index_level = cumprod(1 + portfolio_return)) %>%
    pull(index_level) %>%
    last()

  tibble(
    symbol      = label,
    final_value = final_value,
    sd_return   = sd(pr$portfolio_return)
  )
}))

result_table <- bind_rows(
  left_join(final_values, stats_assets, by = "symbol"),
  stats_portfolios
) %>%
  mutate(
    ci_low  = final_value - ci_mult * sd_return,
    ci_high = final_value + ci_mult * sd_return
  ) %>%
  select(symbol, final_value, ci_low, ci_high)

print(result_table)
```

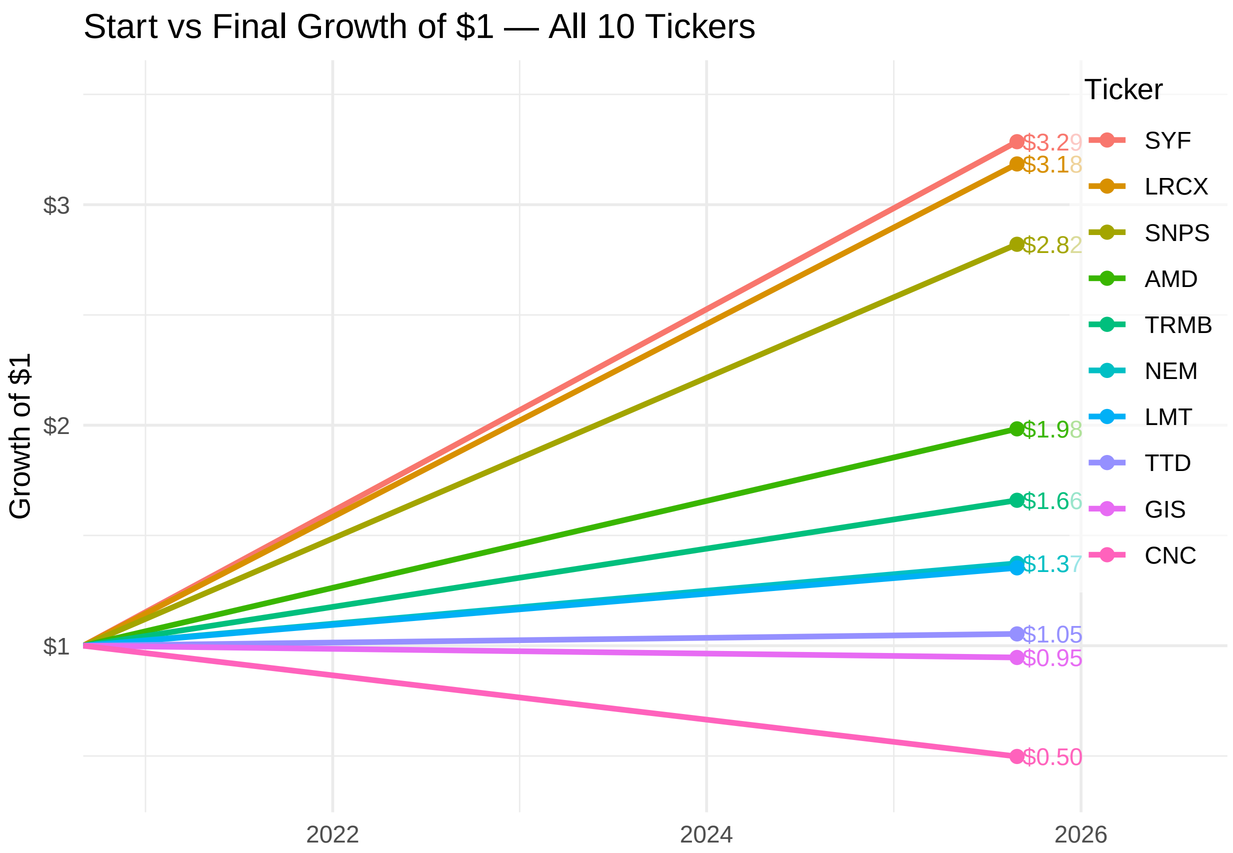
```
## # A tibble: 4 x 4
##   symbol                   final_value ci_low ci_high
##   <chr>                          <dbl>  <dbl>   <dbl>
## 1 GIS                            0.947  0.853    1.04
## 2 LRCX                           3.18   2.97     3.40
## 3 50/50 LRCX-GIS                 1.96   1.85     2.07
## 4 19.90% LRCX / 80.10% GIS       1.30   1.22     1.38
```

```r
library(scales)

base_rows_all <- tibble(
  date = price_start,
  symbol = tickers,
  monthly_return = NA_real_,
  index_level = 1
)

cum_returns_all <- monthly_returns %>%
  filter(symbol %in% tickers) %>%
  arrange(symbol, date) %>%
  group_by(symbol) %>%
  mutate(index_level = cumprod(1 + monthly_return)) %>%
  ungroup() %>%
  bind_rows(base_rows_all) %>%
  arrange(symbol, date) %>%
  select(date, symbol, index_level)

last_values_all <- cum_returns_all %>%
  arrange(date) %>%
  group_by(symbol) %>%
  summarise(final_value = last(index_level), .groups = "drop") %>%
  arrange(desc(final_value))

plot_data_all <- cum_returns_all %>%
  mutate(symbol = factor(symbol, levels = last_values_all$symbol))

label_data_all <- plot_data_all %>%
  group_by(symbol) %>%
  filter(date == max(date)) %>%
  ungroup() %>%
  mutate(label = paste0("$", formatC(index_level, format = "f", digits = 2)))

color_values_all <- setNames(scales::hue_pal()(length(tickers)), last_values_all$symbol)

ggplot(plot_data_all, aes(x = date, y = index_level, color = symbol)) +
  geom_line(linewidth = 1) +
  geom_text(
    data = label_data_all,
    aes(label = label),
    hjust = -0.05,
    vjust = 0.5,
    size = 3,
    show.legend = FALSE,
    check_overlap = TRUE
  ) +
  scale_color_manual(values = color_values_all, limits = levels(plot_data_all$symbol)) +
  scale_x_date(expand = expansion(mult = c(0, 0.12))) +
  scale_y_continuous(labels = dollar_format(prefix = "$")) +
  labs(
    title = "Cumulative Growth of $1 - All 10 Tickers (Sep 2020 onward)",
    y = "Growth of $1",
    x = NULL,
    color = "Ticker"
  ) +
  theme_minimal() +
  theme(
    legend.position = c(0, 1),
    legend.justification = c(0, 1),
    legend.background = element_rect(fill = alpha("white", 0.6), color = NA)
  )
```

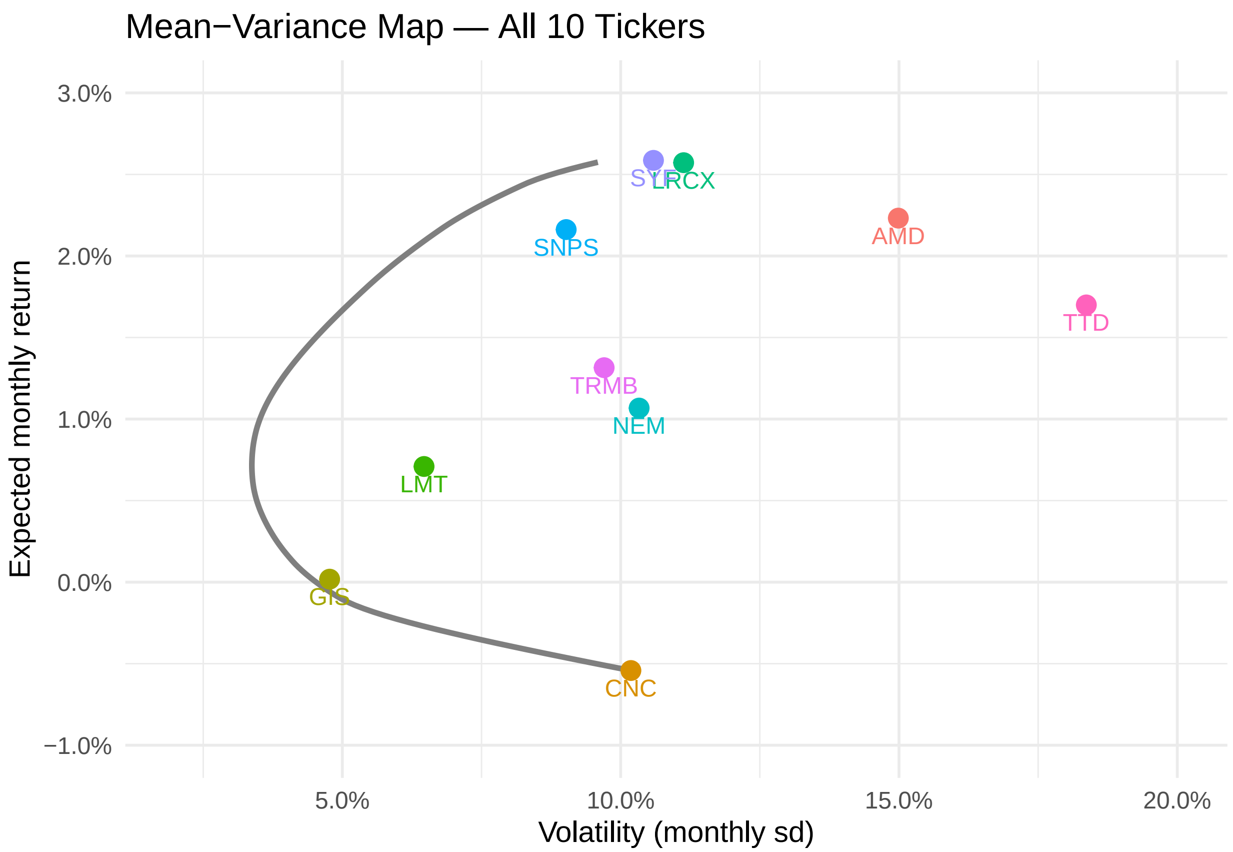


```
end_date <- max(cum_returns_all$date)

segment_data <- last_values_all %>%
  mutate(
    start_date  = price_start,
    start_value = 1,
    end_date    = end_date
  )

segment_plot <- ggplot() +
  geom_segment(
    data = segment_data,
    aes(
      x    = start_date,
      xend = end_date,
      y    = start_value,
      yend = final_value,
      color = symbol
    ),
    linewidth = 1
  ) +
  geom_point(
    data = segment_data,
```

```r
      aes(x = end_date, y = final_value, color = symbol),
      size = 2
    ) +
    geom_text(
      data = segment_data,
      aes(
        x = end_date,
        y = final_value,
        label = paste0("$", formatC(final_value, format = "f", digits = 2)),
        color = symbol
      ),
      nudge_x = 70,              # desplaza texto ~1 mes a la derecha
      vjust = 0.5,
      size = 3,
      show.legend = FALSE,
      check_overlap = TRUE
    ) +
    scale_color_manual(values = color_values_all, limits = last_values_all$symbol) +
    scale_x_date(expand = expansion(mult = c(0, 0.18))) +
    scale_y_continuous(
      limits = c(0.4, 3.5),
      labels = dollar_format(prefix = "$")
    ) +
    labs(
      title = "Start vs Final Growth of $1 - All 10 Tickers",
      x = NULL,
      y = "Growth of $1",
      color = "Ticker"
    ) +
    theme_minimal() +
  theme(
    legend.position = c(1, 1),
    legend.justification = c(1, 1),
    legend.background = element_rect(fill = alpha("white", 0.6), color = NA)
  )

segment_plot
```



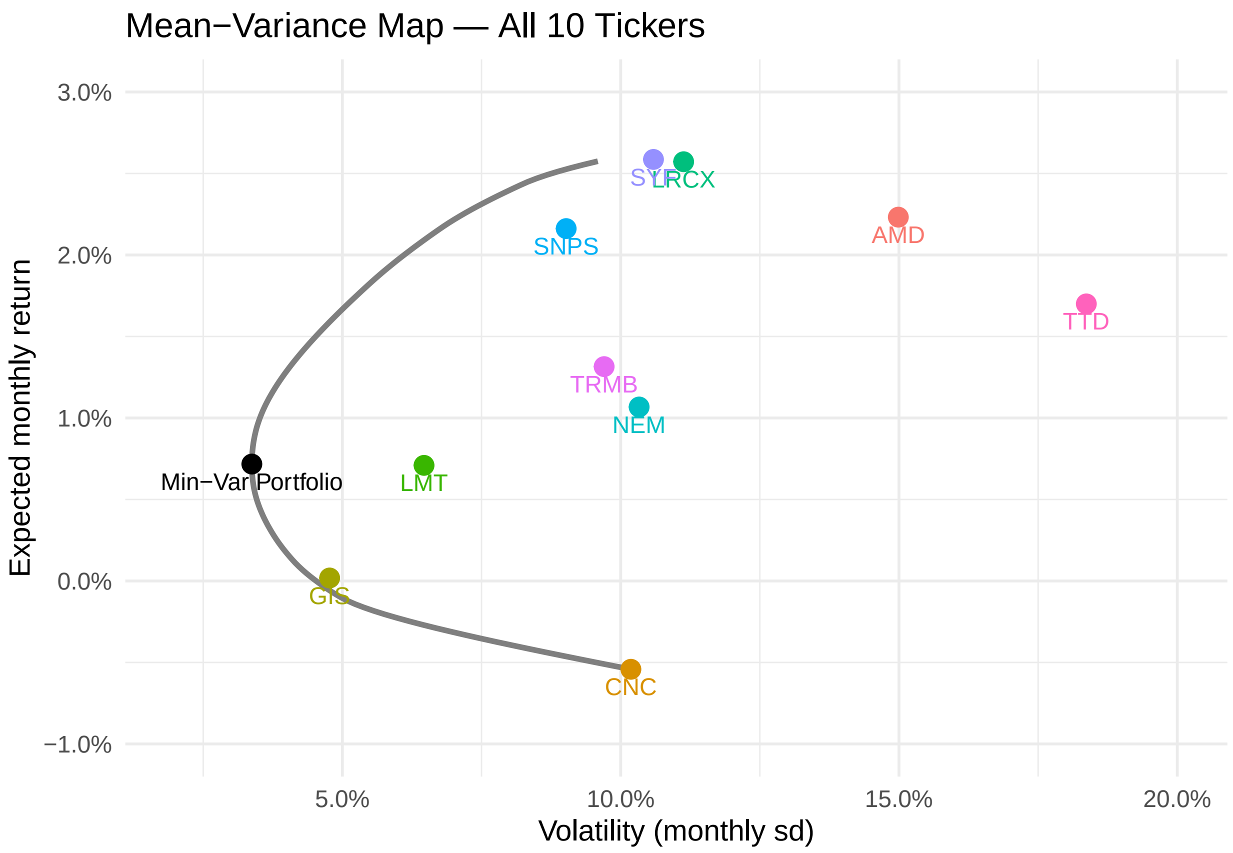
```r
library(quadprog)

returns_wide_all <- monthly_returns %>%
  filter(symbol %in% tickers) %>%
  select(date, symbol, monthly_return) %>%
  pivot_wider(names_from = symbol, values_from = monthly_return) %>%
  arrange(date)

returns_matrix <- returns_wide_all %>%
  select(-date) %>%
  as.matrix()

mu_vec <- colMeans(returns_matrix, na.rm = TRUE)
cov_mat_all <- cov(returns_matrix, use = "pairwise.complete.obs")

stats_all <- tibble(
  symbol = names(mu_vec),
  expected_return = mu_vec,
  sd_return = apply(returns_matrix, 2, sd, na.rm = TRUE)
)

target_returns <- seq(
  from = min(mu_vec),
  to   = max(mu_vec) + .0001,    # empuja la frontera hacia retornos más altos
  length.out = 150
)

Amat_base <- rbind(
    rep(1, length(mu_vec)),
    mu_vec,
    diag(length(mu_vec))
  )

  frontier_points <- map_dfr(target_returns, function(tr) {
    bvec <- c(1, tr, rep(0, length(mu_vec)))
    sol <- tryCatch(
      solve.QP(2 * cov_mat_all, rep(0, length(mu_vec)), t(Amat_base), bvec, meq = 2),
      error = function(e) NULL
    )
    if (is.null(sol)) return(NULL)

    tibble(
      expected_return = tr,
      sd_return = sqrt(as.numeric(t(sol$solution) %*% cov_mat_all %*% sol$solution))
    )
}) %>% drop_na()

  color_values_all <- setNames(scales::hue_pal()(length(tickers)), stats_all$symbol)

ggplot() +
    geom_path(
      data = frontier_points,
      aes(x = sd_return, y = expected_return),
      color = "grey50",
      linewidth = 1
    ) +
    geom_point(
      data = stats_all,
      aes(x = sd_return, y = expected_return, color = symbol),
      size = 3
    ) +
    geom_text(
      data = stats_all,
      aes(x = sd_return, y = expected_return, label = symbol, color = symbol),
      vjust = 1.5,
      size = 3,
      show.legend = FALSE
    ) +
    scale_color_manual(values = color_values_all, limits = stats_all$symbol) +
    scale_x_continuous(
      limits = c(0.02, 0.2),
      labels = scales::percent_format(accuracy = 0.1)
    ) +
    scale_y_continuous(labels = scales::percent_format(accuracy = 0.1),
                             limits = c(-0.01, 0.03)) +
    labs(
      title = "Mean-Variance Map - All 10 Tickers",
      x = "Volatility (monthly sd)",
      y = "Expected monthly return"
    ) +
    guides(color = "none") +
    theme_minimal() +
    theme(legend.position = "none")
```



```r
library(quadprog)

assets_all <- tickers
n_assets <- length(assets_all)

Dmat <- 2 * cov_mat_all
dvec <- rep(0, n_assets)

Amat <- cbind(
  rep(1, n_assets),         # suma de pesos = 1  (meq = 1)
  diag(n_assets)            # pesos   0
)
bvec <- c(1, rep(0, n_assets))

qp_mv <- solve.QP(Dmat, dvec, Amat, bvec, meq = 1)
weights_mv <- setNames(qp_mv$solution, assets_all)

min_var_weights <- tibble(
  symbol = assets_all,
  weight = round(weights_mv, 4)
) %>%
  arrange(desc(weight))

print(min_var_weights)
```

```
## # A tibble: 10 x 2
##    symbol weight
##    <chr>   <dbl>
##  1 GIS    0.540
##  2 SNPS   0.183
##  3 LMT    0.112
##  4 TRMB   0.0462
##  5 NEM    0.0452
##  6 SYF    0.0305
##  7 TTD    0.03
##  8 CNC    0.013
##  9 AMD    0
## 10 LRCX   0
```

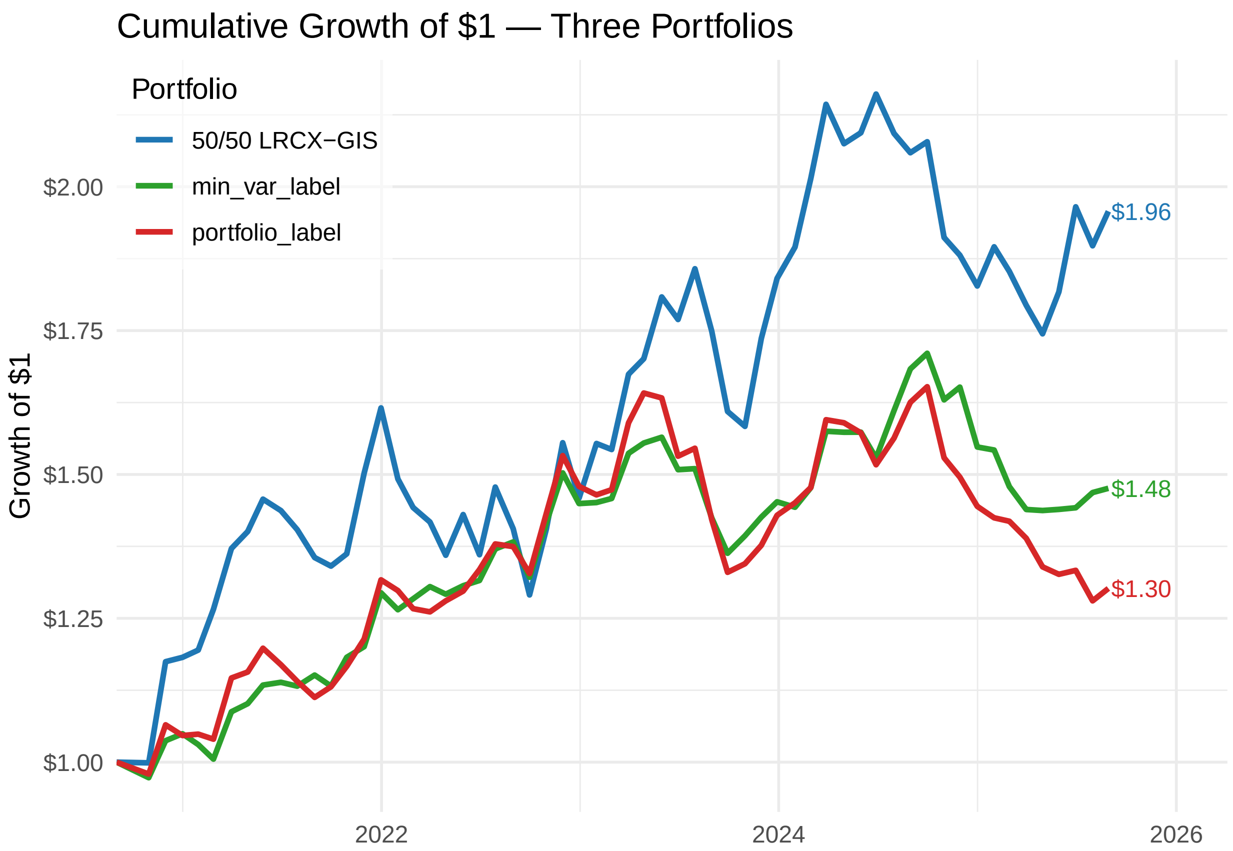
```r
library(quadprog)

returns_wide_all <- monthly_returns %>%
  filter(symbol %in% tickers) %>%
  select(date, symbol, monthly_return) %>%
  pivot_wider(names_from = symbol, values_from = monthly_return) %>%
  arrange(date)

returns_matrix <- returns_wide_all %>%
  select(-date) %>%
  as.matrix()

mu_vec <- colMeans(returns_matrix, na.rm = TRUE)
cov_mat_all <- cov(returns_matrix, use = "pairwise.complete.obs")

stats_all <- tibble(
  symbol = names(mu_vec),
  expected_return = mu_vec,
  sd_return = apply(returns_matrix, 2, sd, na.rm = TRUE)
)

target_returns <- seq(
  from = min(mu_vec),
  to   = max(mu_vec) + 0.0001,
  length.out = 150
)

Amat_base <- rbind(
  rep(1, length(mu_vec)),
  mu_vec,
  diag(length(mu_vec))
)

frontier_points <- map_dfr(target_returns, function(tr) {
  bvec <- c(1, tr, rep(0, length(mu_vec)))
  sol <- tryCatch(
    solve.QP(2 * cov_mat_all, rep(0, length(mu_vec)), t(Amat_base), bvec, meq = 2),
    error = function(e) NULL
  )
  if (is.null(sol)) return(NULL)

  tibble(
    expected_return = tr,
    sd_return = sqrt(as.numeric(t(sol$solution) %*% cov_mat_all %*% sol$solution))
  )
}) %>% drop_na()

weights_mv <- solve.QP(
  2 * cov_mat_all,
  rep(0, length(mu_vec)),
  cbind(rep(1, length(mu_vec)), diag(length(mu_vec))),
  c(1, rep(0, length(mu_vec))),
  meq = 1
)$solution

min_var_point <- tibble(
  symbol = "Min-Var Portfolio",
  expected_return = sum(weights_mv * mu_vec),
  sd_return = sqrt(as.numeric(t(weights_mv) %*% cov_mat_all %*% weights_mv))
)

stats_all_ext <- bind_rows(stats_all, min_var_point)

color_values_all <- setNames(scales::hue_pal()(length(tickers)), stats_all$symbol)
color_values_ext <- c(color_values_all, "Min-Var Portfolio" = "black")

ggplot() +
  geom_path(
    data = frontier_points,
    aes(x = sd_return, y = expected_return),
    color = "grey50",
    linewidth = 1
  ) +
  geom_point(
    data = stats_all_ext,
    aes(x = sd_return, y = expected_return, color = symbol),
    size = 3
  ) +
  geom_text(
    data = stats_all_ext,
    aes(x = sd_return, y = expected_return, label = symbol, color = symbol),
    vjust = 1.5,
    size = 3,
    show.legend = FALSE
  ) +
  scale_color_manual(values = color_values_ext, limits = stats_all_ext$symbol) +
  scale_x_continuous(
    limits = c(0.02, 0.2),
    labels = scales::percent_format(accuracy = 0.1)
  ) +
  scale_y_continuous(
    limits = c(-0.01, 0.03),
    labels = scales::percent_format(accuracy = 0.1)
  ) +
  labs(
    title = "Mean-Variance Map - All 10 Tickers",
    x = "Volatility (monthly sd)",
    y = "Expected monthly return"
  ) +
  guides(color = "none") +
  theme_minimal() +
  theme(legend.position = "none")
```



```r
asset_stats <- monthly_returns %>%
  filter(symbol %in% tickers) %>%
  group_by(symbol) %>%
  summarise(
    expected_return = mean(monthly_return),
    sd_return       = sd(monthly_return),
    sharpe_ratio    = expected_return / sd_return,
    .groups = "drop"
  )

weights_50_50 <- c(LRCX = 0.5, GIS = 0.5)
min_var_label <- "Min-Var 10-Asset"
weights_mv_all <- setNames(round(weights_mv, 4), tickers)  # del bloque previo

portfolio_specs <- list(
  "50/50 LRCX-GIS" = weights_50_50,
  portfolio_label  = custom_weights
)
portfolio_specs[[min_var_label]] <- weights_mv_all


portfolio_stats_ext <- imap_dfr(portfolio_specs, function(w, label) {
  returns <- monthly_returns %>%
    filter(symbol %in% names(w)) %>%
    mutate(weight = w[symbol]) %>%
    group_by(date) %>%
    summarise(portfolio_return = sum(monthly_return * weight), .groups = "drop")

  tibble(
    symbol          = label,
    expected_return = mean(returns$portfolio_return),
    sd_return       = sd(returns$portfolio_return),
    sharpe_ratio    = expected_return / sd_return
  )
})

table_levels <- c(tickers, names(portfolio_specs))

final_table <- bind_rows(asset_stats, portfolio_stats_ext) %>%
  mutate(symbol = factor(symbol, levels = table_levels)) %>%
  arrange(symbol)

print(final_table)
```

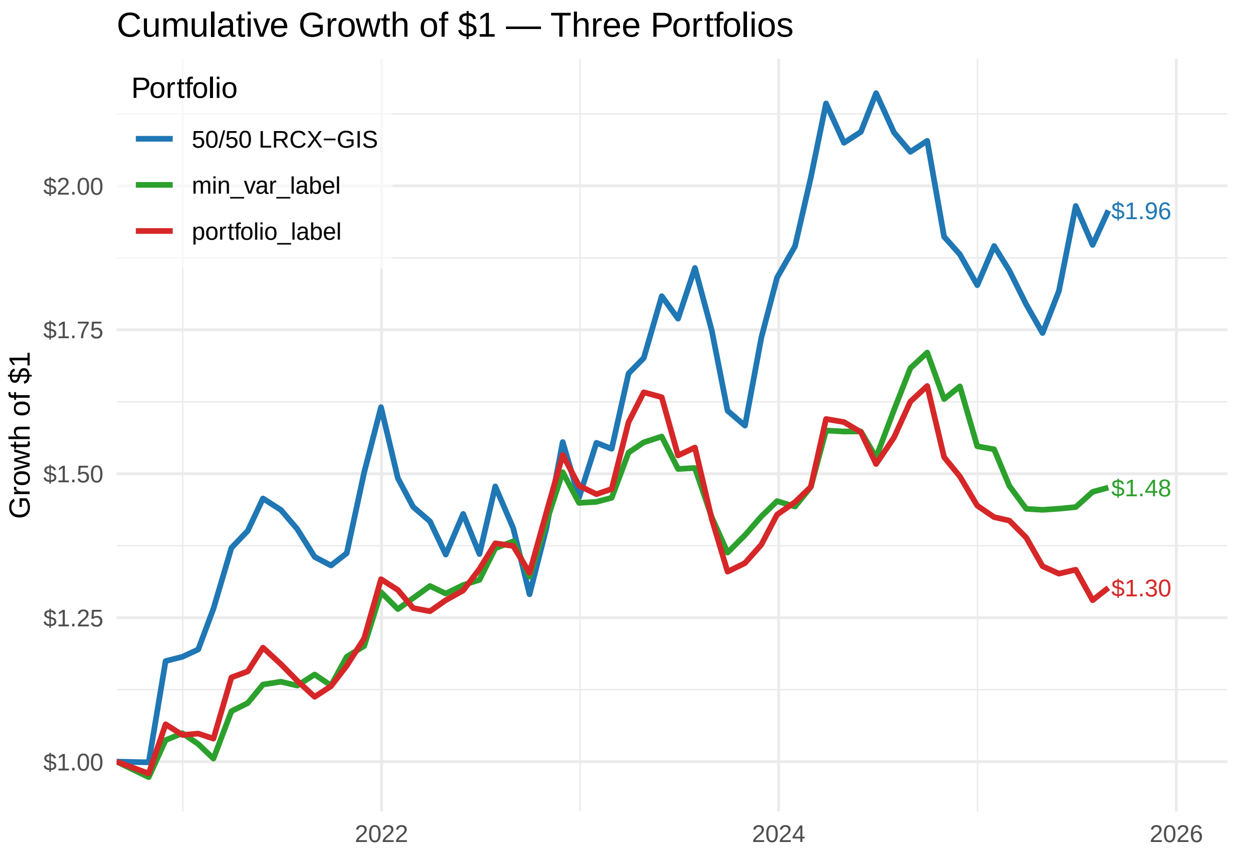
```
## # A tibble: 13 x 4
##    symbol           expected_return sd_return sharpe_ratio
##    <fct>                      <dbl>     <dbl>        <dbl>
##  1 AMD                     0.0223      0.150       0.149
##  2 CNC                    -0.00542     0.102      -0.0532
##  3 GIS                     0.000182    0.0477      0.00382
##  4 LMT                     0.00709     0.0647      0.110
##  5 LRCX                    0.0257      0.111       0.231
##  6 NEM                     0.0107      0.103       0.103
##  7 SNPS                    0.0216      0.0902      0.240
##  8 SYF                     0.0259      0.106       0.244
##  9 TRMB                    0.0132      0.0970      0.136
## 10 TTD                     0.0170      0.184       0.0926
## 11 50/50 LRCX-GIS          0.0130      0.0560      0.231
## 12 portfolio_label         0.00526     0.0401      0.131
## 13 Min-Var 10-Asset        0.00717     0.0337      0.213
```

```r
portfolio_defs <- list(
  "50/50 LRCX-GIS"       = weights_50_50,
  portfolio_label        = custom_weights,
  min_var_label          = weights_mv_all
)

portfolio_cum_returns <- imap_dfr(portfolio_defs, function(w, label) {
  returns <- monthly_returns %>%
    filter(symbol %in% names(w)) %>%
    mutate(weight = w[symbol]) %>%
    group_by(date) %>%
    summarise(portfolio_return = sum(monthly_return * weight), .groups = "drop") %>%
    arrange(date) %>%
    mutate(index_level = cumprod(1 + portfolio_return)) %>%
    mutate(symbol = label) %>%
    select(date, symbol, index_level)

  bind_rows(
    tibble(date = price_start, symbol = label, index_level = 1),
    returns
  )
})

last_values_port <- portfolio_cum_returns %>%
  arrange(date) %>%
  group_by(symbol) %>%
  summarise(final_value = last(index_level), .groups = "drop") %>%
  arrange(desc(final_value))

portfolio_colors <- c(
  "50/50 LRCX-GIS" = "#1f77b4",
  portfolio_label  = "#d62728",
  min_var_label    = "#2ca02c"
)

plot_portfolios <- portfolio_cum_returns %>%
  mutate(symbol = factor(symbol, levels = last_values_port$symbol))

label_data_port <- plot_portfolios %>%
  group_by(symbol) %>%
  filter(date == max(date)) %>%
  ungroup() %>%
  mutate(label = paste0("$", formatC(index_level, format = "f", digits = 2)))

ggplot(plot_portfolios, aes(x = date, y = 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, limits = last_values_port$symbol) +
  scale_x_date(expand = expansion(mult = c(0, 0.12))) +
  scale_y_continuous(labels = scales::dollar_format(prefix = "$")) +
  labs(
    title = "Cumulative Growth of $1 - Three Portfolios",
    y = "Growth of $1",
    x = NULL,
    color = "Portfolio"
  ) +
  theme_minimal() +
  theme(
    legend.position = c(0, 1),
    legend.justification = c(0, 1),
    legend.background = element_rect(fill = scales::alpha("white", 0.6), color = NA)
  )
```

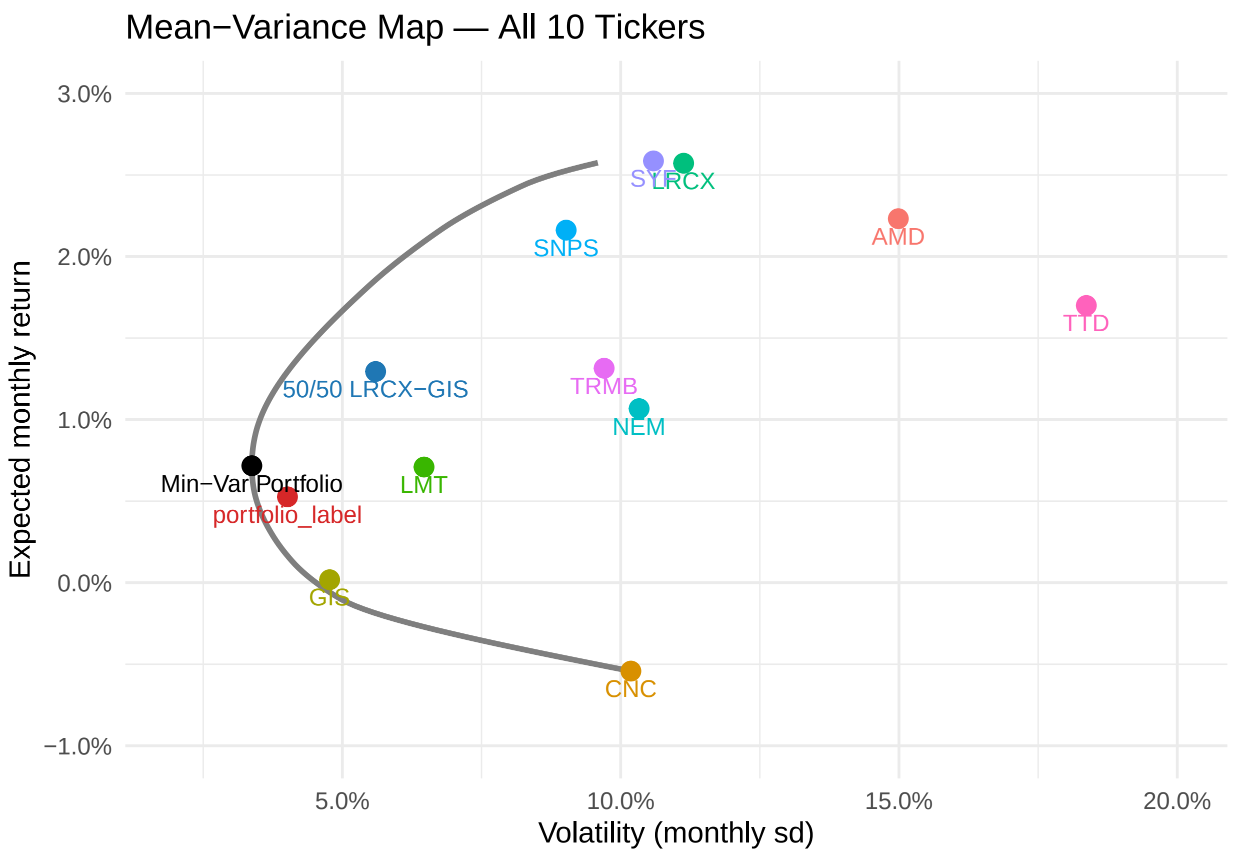


```
weights_50_50 <- c(LRCX = 0.5, GIS = 0.5)  # ya definido antes
custom_weights <- c(LRCX = 0.1990055, GIS = 0.8009945)  # o el vector que uses
portfolio_label <- "19.90% LRCX / 80.10% GIS"           # etiqueta actual

portfolio_points <- tribble(
  ~symbol,                 ~weights,
  "50/50 LRCX-GIS",        weights_50_50,
  portfolio_label,         custom_weights
) %>%
  mutate(
    expected_return = map_dbl(weights, ~ sum(.x * mu_vec[names(.x)])),
    sd_return       = map_dbl(weights,  ~ sqrt(as.numeric(t(.x) %*% cov_mat_all[names(.x), names(.x)] %*% .x)))
  ) %>%
  select(-weights)

stats_all_ext <- bind_rows(stats_all, portfolio_points, min_var_point)

color_values_ext <- c(
  color_values_all,
  "50/50 LRCX-GIS"          = "#1f77b4",
  portfolio_label           = "#d62728",
  "Min-Var Portfolio"       = "black"
)
```

```r


  ggplot(plot_portfolios, aes(x = date, y = 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, limits = last_values_port$symbol) +
    scale_x_date(expand = expansion(mult = c(0, 0.12))) +
    scale_y_continuous(labels = scales::dollar_format(prefix = "$")) +
    labs(
      title = "Cumulative Growth of $1 - Three Portfolios",
      y = "Growth of $1",
      x = NULL,
      color = "Portfolio"
    ) +
    theme_minimal() +
    theme(
      legend.position = c(0, 1),
      legend.justification = c(0, 1),
      legend.background = element_rect(fill = scales::alpha("white", 0.6), color = NA)
    )
```



```r
library(quadprog)

weights_50_50 <- c(LRCX = 0.5, GIS = 0.5)
portfolio_label <- "19.90% LRCX / 80.10% GIS"   # usa el texto que ya venías ocupando
custom_weights <- c(LRCX = 0.1990055, GIS = 0.8009945)

portfolio_points <- list(
  "50/50 LRCX-GIS" = weights_50_50,
  portfolio_label  = custom_weights
) %>%
  imap_dfr(function(w, label) {
    assets <- names(w)
    mu     <- mu_vec[assets]
    covar  <- cov_mat_all[assets, assets]

    tibble(
      symbol          = label,
      expected_return = sum(w * mu),
      sd_return       = sqrt(as.numeric(t(w) %*% covar %*% w))
    )
  })

  stats_all_ext <- bind_rows(stats_all, portfolio_points, min_var_point)

color_values_ext <- c(
  color_values_all,
  "50/50 LRCX-GIS"        = "#1f77b4",
  portfolio_label         = "#d62728",
  "Min-Var Portfolio"     = "black"
)

ggplot() +
  geom_path(
    data = frontier_points,
    aes(x = sd_return, y = expected_return),
    color = "grey50",
    linewidth = 1
  ) +
  geom_point(
    data = stats_all_ext,
    aes(x = sd_return, y = expected_return, color = symbol),
    size = 3
  ) +
  geom_text(
    data = stats_all_ext,
    aes(x = sd_return, y = expected_return, label = symbol, color = symbol),
    vjust = 1.5,
    size = 3,
    show.legend = FALSE
  ) +
  scale_color_manual(values = color_values_ext, limits = stats_all_ext$symbol) +
  scale_x_continuous(
    limits = c(0.02, 0.2),
    labels = scales::percent_format(accuracy = 0.1)
  ) +
  scale_y_continuous(
    limits = c(-0.01, 0.03),
    labels = scales::percent_format(accuracy = 0.1)
  ) +
  labs(
    title = "Mean-Variance Map - All 10 Tickers",
    x = "Volatility (monthly sd)",
    y = "Expected monthly return"
  ) +
  guides(color = "none") +
  theme_minimal() +
  theme(legend.position = "none")
```



## 3 Conclusión.

- Los rendimientos logarítmicos y porcentuales generan resultados prácticamente equivalentes.
- El enfoque no condicional, asume una volatilidad constante en el tiempo y, por tanto, no captura la naturaleza dinámica y heterocedástica observada en los mercados financieros.