ACT974 Financial Econometrics Term paper

Comparing GARCH and Stochastic Volatility Models with and without Realized Volatility

Due: 16 December 2025

Gurmanjot Singh

Instructor: Professor Tony Wirjanto

Department of Statistics and Actuarial Sciences

University of Waterloo

Abstract

Generalized Autoregressive Conditional Heteroskedasticity (GARCH) models and stochastic volatility (SV) models are two dominant frameworks for modeling time-varying volatility in financial returns. Although SV models offer greater flexibility through latent volatility dynamics, empirical work often finds that simple GARCH specifications remain competitive when estimation relies only on daily returns. This paper examines how the relative performance of GARCH and SV models changes when realized volatility (RV) measures constructed from high-frequency data are incorporated as an additional source of volatility information.

Using daily S&P 500 returns and realized volatility estimates over 2000–2013, we compare four specifications: returns-only GARCH(1,1), returns-only SV, Realized GARCH with a measurement equation for RV, and a realized-measure SV model with an explicit RV observation equation. All models are evaluated under a common out-of-sample, one-step-ahead forecasting design using realized volatility as a proxy for latent variance.

In the returns-only setting, the SV model produces slightly lower but broadly comparable RV forecast losses than GARCH across QLIKE, MSE, and MAE. When realized volatility is incorporated, both realized-measure models substantially improve forecast accuracy relative to returns-only models, but their ranking depends on the loss function: Realized GARCH attains the lowest QLIKE, while the realized-measure SV model attains the lowest MSE and MAE. Interpreting these metrics jointly, the results suggest that Realized GARCH is marginally preferred under the proxy-robust QLIKE criterion, whereas realized-measure SV delivers smaller average forecast errors in squared and absolute loss. Overall, incorporating realized measures materially changes the empirical comparison between GARCH and SV, and provides a practical route to improving predictive distributions for returns through better volatility inference. In this setting, the Realized GARCH estimates further indicate that, when realized volatility is highly informative, the traditional GARCH variance recursion becomes empirically redundant, with volatility dynamics driven primarily by the realized measure rather than by past returns.

Contents

List of Figures

List of Tables

1 Introduction

Modeling and forecasting time-varying volatility is a central problem in financial econometrics, with direct applications to risk management, asset pricing, and derivative valuation. Empirical return series for financial assets exhibit volatility clustering, excess kurtosis, and time-varying conditional variance, motivating the development of models that depart from constant-variance assumptions [4].

Two dominant frameworks have emerged for representing financial volatility. The first is the class of Generalized Autoregressive Conditional Heteroskedasticity (GARCH) models, in which conditional variance evolves deterministically as a function of past returns and past volatility [4]. The second is the stochastic volatility (SV) framework, which treats volatility as an unobserved latent process driven by its own stochastic dynamics [13, 6]. Both approaches are widely studied and form a core component of modern financial econometrics.

From a modeling perspective, stochastic volatility models provide a more flexible representation of volatility dynamics by allowing for autonomous volatility innovations. Nevertheless, a substantial empirical literature finds that standard GARCH models perform comparably to, and in some cases outperform, stochastic volatility models when estimation relies solely on daily return data [8]. This apparent discrepancy between model flexibility and empirical performance raises questions about the information content of returns for volatility inference.

One explanation lies in the limited observability of volatility when only low-frequency returns are available. Daily returns provide an indirect and noisy signal of the underlying volatility process, making it difficult to distinguish between large return shocks and genuine changes in volatility. The availability of high-frequency intraday data has led to the construction of realized volatility measures, which provide consistent but noisy estimates of integrated volatility over fixed time intervals [1]. Realized volatility therefore offers a means of improving volatility inference by partially observing the latent volatility process.

Importantly, incorporating realized volatility does not eliminate the structural differences between GARCH and stochastic volatility models. In GARCH-type specifications, realized measures typically enter as exogenous inputs in deterministic variance recursions, whereas in stochastic volatility models they are treated as noisy observations of an underlying latent state. As a result, the role of realized volatility in estimation and forecasting differs fundamentally across frameworks, and improvements in forecast performance need not be uniform across loss functions or forecast objectives. This makes a direct empirical comparison between GARCH and stochastic volatility models with realized measures nontrivial, even when models are evaluated on the same dataset.

The objective of this paper is to examine how the relative performance of GARCH and stochastic volatility models changes once realized volatility is incorporated into the modeling framework. We first compare standard GARCH and stochastic volatility models estimated using daily returns only. We then extend both frameworks by incorporating realized volatility, using GARCH models with realized volatility inputs [9] and stochastic volatility models with measurement equations for realized volatility [12, 11]. This setting allows us to assess how the inclusion of realized volatility alters the empirical comparison between the two model classes, and whether different loss functions favor deterministic or latent volatility dynamics.

This paper proceeds by first reviewing the relevant literature on GARCH models, stochastic volatility models, and realized volatility measures, with particular emphasis on how these approaches differ in their treatment of volatility dynamics. We then present the theoretical specifications of the competing models and discuss their estimation strategies and practical considerations. Using a standard benchmark equity dataset with available realized volatility, we conduct an empirical comparison between GARCH and stochastic volatility models in both returns-only and realized-volatility-augmented settings. The resulting estimates and forecasting performance are analyzed to assess how the inclusion of realized volatility affects the relative performance of the two model classes. Finally, we discuss the implications of the findings for volatility modeling and conclude with directions for future research.

2 Literature Review and Theoretical Background

This section reviews the principal theoretical frameworks used to model time-varying volatility in financial returns and examines how the availability of realized volatility measures alters the volatility inference problem. We focus on three closely related strands of the literature. The first consists of GARCH-type models, which update conditional variance deterministically as a function of past returns and past volatility. The second is the class of stochastic volatility (SV) models, which represent volatility as a latent stochastic process evolving independently of observed returns. The third strand concerns realized volatility measures constructed from high-frequency data, which provide noisy but informative proxies for integrated variance.

The discussion draws on the foundational GARCH framework of Bollerslev [4], early stochastic volatility formulations and surveys by Taylor [13] and Ghysels et al. [6], and realized volatility theory developed by Andersen et al. [1]. Empirical benchmark results from Hansen and Lunde [8] motivate the comparison of returns-only models, while joint modeling approaches incorporating realized measures are drawn from Realized GARCH [9] and realized-measure stochastic volatility models [12, 11]. Related forecasting frameworks, including HAR-RV [5] and HEAVY models [10], are discussed to provide context for how realized measures are used in volatility forecasting.

2.1 Stylized Facts and the Volatility Inference Problem

A central empirical motivation for volatility modeling is that financial returns exhibit volatility clustering and persistent conditional heteroskedasticity: large absolute returns tend to be followed by large absolute returns, and the conditional variance varies over time. These features are well documented across asset classes and motivate models in which volatility is allowed to evolve dynamically rather than remaining constant.

Despite its importance, volatility is not directly observed. Daily returns provide only an indirect and noisy signal of the underlying variance process. A large return realization may reflect either a genuinely high level of conditional variance or an unusually large innovation, and this confounding limits the information content of returns for identifying volatility dynamics. This limitation is particularly important when comparing model classes that differ in how volatility evolves, because the data may be insufficiently informative to discriminate between deterministic variance updating and latent stochastic dynamics.

This issue is central to the empirical findings of Hansen and Lunde [8], who show that simple volatility models are often difficult to outperform in out-of-sample forecasting exercises. Their results emphasize that empirical rankings depend on both the volatility proxy used for evaluation and the chosen loss function. When volatility is weakly observed, model flexibility does not necessarily translate into superior forecast performance, and differences in model structure may only become apparent when additional information is introduced.

2.2 GARCH Models as Deterministic Conditional Variance Updating

The GARCH model introduced by Bollerslev [4] generalizes the ARCH framework by allowing conditional variance to depend on both past squared innovations and past variances. A standard benchmark specification is the GARCH(1,1) model,

$$r_t = \mu + \epsilon_t, \qquad \epsilon_t = \sqrt{h_t}\, z_t, \qquad z_t \sim (0,1), \tag{2.1}$$

$$h_t = \omega + \alpha\epsilon_{t-1}^2 + \beta h_{t-1}, \tag{2.2}$$

where h_t denotes the conditional variance and (ω, α, β) are parameters satisfying positivity restrictions and, under standard conditions, covariance stationarity when $\alpha + \beta < 1$.

The variance recursion in (2.2) implies that volatility dynamics are entirely driven by observed past returns. The parameter α governs the immediate impact of return shocks on volatility, while β controls the persistence of volatility over time. When $\alpha + \beta$ is close to unity, volatility exhibits strong persistence, a feature commonly observed in financial data. Importantly, there is no independent innovation to volatility: conditional on past returns, the future variance path is fully determined.

This deterministic updating structure can be interpreted as a reduced-form mechanism for learning about volatility from return realizations. While this approach is computationally convenient and often empirically effective, it implicitly rules out volatility movements that are not directly associated with large return shocks. As a result, changes in volatility must be justified ex post by observed return behavior.

Empirically, GARCH models have proven difficult to outperform in returns-only forecasting comparisons. The study of Hansen and Lunde [8] provides a prominent benchmark, demonstrating that simple GARCH specifications often perform comparably to more complex alternatives under a range of loss functions. An important implication of this result is that empirical performance depends not only on model structure but also on the information available for volatility inference and the criteria used for evaluation.

2.3 Stochastic Volatility Models as Latent-State Dynamics

Stochastic volatility models, developed in early form by Taylor [13] and surveyed by Ghysels et al. [6], represent volatility as an unobserved latent process evolving stochastically over time. A canonical specification is given by

$$r_t = \mu + \exp(h_t/2)\, z_t, \qquad z_t \sim (0,1), \tag{2.3}$$

$$h_t = \mu_h + \phi(h_{t-1} - \mu_h) + \eta_t, \qquad \eta_t \sim \mathcal{N}(0, \sigma_\eta^2), \tag{2.4}$$

where h_t denotes log-volatility following an AR(1) process.

In contrast to GARCH models, volatility dynamics in SV models are driven by an autonomous innovation η_t. Even after conditioning on past returns, future volatility remains uncertain. This feature allows SV models to capture gradual volatility shifts that are not necessarily preceded by large return realizations, providing a richer probabilistic description of volatility dynamics.

The flexibility of SV models comes at the cost of more involved inference. Because volatility is latent, likelihood evaluation requires integrating over the unobserved state process. As discussed by Ghysels et al. [6] and Asai et al. [2], estimation is commonly performed using Bayesian MCMC methods, particle filtering, or approximate likelihood techniques. When only daily returns are available, the latent volatility state is weakly identified, which can lead to substantial estimation uncertainty and slow convergence in simulation-based methods.

These inference challenges provide one explanation for why SV models do not always dominate GARCH models in returns-only empirical comparisons. Although SV models are theoretically more flexible, the limited information content of daily returns may prevent this flexibility from translating into superior forecast performance.

2.4 Realized Volatility and High-Frequency Measurement

The availability of high-frequency intraday data has led to the development of realized volatility measures that provide direct, though noisy, estimates of integrated variance. Andersen et al. [1] formalize the use of realized variance, constructed as the sum of squared intraday returns,

$$RV_t = \sum_{i=1}^{M_t} r_{t,i}^2, \tag{2.5}$$

as a consistent estimator of quadratic variation under suitable conditions.

By aggregating high-frequency information, realized volatility measures substantially increase the information available for volatility inference relative to daily returns alone. However, these measures are subject to market microstructure noise, jumps, and finite-sample effects. Barndorff-Nielsen and Shephard [3] demonstrate that robustness considerations are central in realized volatility measurement, motivating the treatment of realized measures as noisy observations rather than exact measures of latent variance.

From a modeling perspective, this measurement error plays a crucial role. Models that incorporate realized volatility must balance responsiveness to informative signals against the risk of overreacting to noise. This trade-off has important implications for both estimation and forecast evaluation.

2.5 Joint Models Using Returns and Realized Measures

One approach to incorporating realized volatility is the Realized GARCH framework proposed by Hansen et al. [9]. In this class of models, the conditional variance follows a GARCH-type recursion augmented by realized volatility inputs, and a measurement equation links the realized measure to latent variance,

$$r_t = \mu + \sqrt{h_t}\, z_t, \tag{2.6}$$

$$h_t = \omega + \beta h_{t-1} + \gamma x_{t-1}, \tag{2.7}$$

$$\log x_t = \xi + \delta \log h_t + u_t, \tag{2.8}$$

where x_t denotes a realized measure and u_t captures measurement error.

In this framework, volatility updating remains deterministic, but realized measures provide additional conditioning information that improves volatility tracking. Because realized volatility enters the variance recursion directly, Realized GARCH models can respond quickly to changes in observed volatility, which may reduce systematic underestimation under certain loss functions.

An alternative approach incorporates realized measures into stochastic volatility models via explicit observation equations. Takahashi et al. [12] and Shirota and Omori [11] develop SV models in which realized volatility is treated as a noisy observation of latent log-volatility,

$$
\begin{aligned}
r_t &= \mu + \exp(h_t/2)\, z_t, && (2.9)\\
h_t &= \mu_h + \phi(h_{t-1} - \mu_h) + \eta_t, && (2.10)\\
\log x_t &= \alpha + h_t + u_t, && (2.11)
\end{aligned}
$$

with distinct innovations governing volatility dynamics and measurement error.

The key distinction between these approaches lies in how realized measures affect volatility dynamics. In Realized GARCH, realized volatility directly influences the variance recursion, while in realized-measure SV models it informs the latent state through filtering without eliminating stochastic volatility innovations. These structural differences imply different forecasting behavior and motivate empirical comparison under alternative evaluation criteria.

2.6 Comparison with Alternative High-Frequency Frameworks

While this study focuses on the comparison between GARCH and Stochastic Volatility frameworks, it is important to contextualize these against other prominent high-frequency volatility models, specifically the Heterogeneous Autoregressive (HAR-RV) model of [5] and the High-Frequency-Based Volatility (HEAVY) models of [10].

The HAR-RV framework has gained immense popularity due to its simplicity; it models realized volatility as a linear function of past daily, weekly, and monthly realized volatilities. While effective at capturing long-memory features through a cascade structure, HAR-RV is fundamentally a reduced-form regression model. It lacks a distinct latent volatility component, implicitly assuming that the realized measure itself is the state variable of interest. This makes it difficult to disentangle measurement noise from genuine volatility shocks, a distinction that is central to the realized-measure SV framework.

Similarly, HEAVY models extend the GARCH logic by using realized measures to drive the variance dynamics. Like Realized GARCH, HEAVY models are observation-driven; the current filtration is generated entirely by past observables. While this ensures likelihood tractability, it forces the model to react to every spike in the realized measure, potentially over-responding to microstructure noise or transient jumps. In contrast, the realized-measure SV framework treated in this paper is parameter-driven. By modeling volatility as a separate latent process that is only *imperfectly* observed through realized measures, the SV framework theoretically offers a superior mechanism for filtering out measurement error [11]. This study therefore focuses on the Realized GARCH versus Realized SV comparison to strictly isolate the implications of deterministic versus stochastic volatility updating, holding the information set constant.

2.7 Motivation and Focus of the Present Study

The literature suggests that returns-only empirical comparisons often find simple GARCH models to be difficult to outperform, despite the greater flexibility of SV frameworks [8]. At the same time, realized volatility measures provide a substantial increase in information about volatility relative to daily returns alone [1]. Both GARCH and SV models have been extended to incorporate realized measures, but through fundamentally different mechanisms.

Relatively few studies provide a unified head-to-head comparison of GARCH and SV models under identical evaluation designs in both returns-only and realized-measure-augmented settings. The present paper addresses this gap by comparing these frameworks using a common dataset, forecast design, and evaluation criteria, with particular emphasis on how volatility updating and filtering interact with realized measures to shape empirical performance.

3 Methodology and Data

This section describes the data construction, model estimation, forecasting protocol, and evaluation criteria used to compare GARCH and stochastic volatility (SV) models in both returns-only and realized-volatility-augmented settings. The design follows the spirit of common volatility forecasting benchmarks in which models are compared under a common information set, a fixed forecast horizon, and multiple loss functions that depend on a volatility proxy [8]. All procedures are implemented in R; the full code used to download, align, estimate, filter, forecast, and evaluate models is provided in Appendix A.

3.1 Data

The empirical application uses daily log returns on the S&P 500 index over the period 2000–2013. Daily adjusted closing prices are obtained from Yahoo Finance and transformed into log returns $r_t = \log(P_t) - \log(P_{t-1})$.

To proxy latent daily variance, we use realized volatility (RV) estimates constructed from high-frequency intraday returns. Specifically, x_t denotes the realized variance measure provided in the `rvsp500` dataset from the `midasr` package [7]. This series is conceptually related to the realized variance estimator studied by Andersen et al. [1], where daily integrated variance is approximated by summing squared high-frequency intraday returns. Because realized measures are subject to microstructure noise and other measurement effects, they are treated as noisy but informative proxies rather than error-free observations of latent variance [1, 3].

The daily return series and RV series are aligned by calendar date using an inner join, and observations with non-finite values are removed; we also impose $x_t > 0$ so that $\log x_t$ is well-defined. The resulting sample is $\{(r_t, x_t)\}_{t=1}^n$, with $\log x_t$ used in models that specify a log-scale measurement equation. While the dataset ends in 2013 due to the availability of the curated `rvsp500` series in the `midasr` package, the period 2000–2013 covers multiple volatility regimes, including the dot-com aftermath and the 2008 financial crisis. This provides a sufficiently rich testing ground to evaluate the methodological differences between the models, which is the primary focus of this study.

For out-of-sample evaluation, the sample is split into an estimation set and a test set. The code uses a fixed holdout of 20% of observations for testing and 80% for estimation, with a single chronological split date. All forecasts are one-step-ahead forecasts produced recursively through the test period.

3.2 Models

We compare four specifications that combine two model classes (GARCH and SV) with two information settings (returns-only and returns plus realized volatility). For each model, the target object for volatility forecasting is the conditional variance h_t of returns, while x_t is treated as a proxy for latent variance in evaluation and, in two models, also enters the model as an observed realized measure.

3.2.1 Returns-only GARCH(1,1)

The returns-only benchmark is the standard GARCH(1,1) model of **?**], (2.1)–(2.2). Conditional on parameter estimates, one-step-ahead variance forecasts in the test set are generated using the recursion for h_t with lagged returns and lagged forecast variance.

3.2.2 Returns-only stochastic volatility

The returns-only SV model is the canonical latent log-volatility specification in (2.3)–(2.4), as developed in early form by Taylor [13] and surveyed in Ghysels et al. [6]. In this framework, volatility is latent and inference is based on the joint distribution of returns and the latent state sequence. The model is estimated on the training sample using simulation-based methods (Bayesian MCMC), which is standard for SV models due to the intractability of the likelihood after integrating out latent volatility [6, 2].

3.2.3 Realized GARCH

To incorporate realized measures in a GARCH-type framework, we implement a Realized GARCH specification in the spirit of Hansen et al. [9], consisting of a variance recursion augmented by lagged realized variance and a measurement equation linking the realized measure to the latent variance, as in (2.6)–(2.8). In this structure, variance updating remains deterministic conditional on past information, but realized volatility provides an additional volatility signal that enters the evolution of h_t and is simultaneously modeled through the measurement equation.

3.2.4 SV with a realized-volatility observation equation

The realized-measure SV model follows the measurement-error SV frameworks of Takahashi et al. [12] and Shirota and Omori [11], combining the latent SV state dynamics (2.10) with the observation equation (2.11) that treats $\log x_t$ as a noisy measurement of latent log-volatility. This formulation distinguishes innovations in volatility dynamics from measurement error in realized volatility, which is a key modeling motivation in realized-measure SV approaches [12, 11].

3.3 Estimation and Filtering

The estimation strategy mirrors how each model treats volatility as observed or latent.

For the GARCH(1,1) model, parameters are estimated on the training sample by (quasi-)maximum likelihood under Gaussian innovations using `rugarch`. Given fitted parameters, the in-sample conditional variance series is recovered and the one-step-ahead forecast recursion is applied through the test period.

For the Realized GARCH model, parameters are estimated on the training sample by maximizing the joint log-likelihood implied by the return equation and the measurement equation, as in Hansen et al. [9]. In implementation, the measurement equation is specified on the log scale, and numerical optimization is performed with positivity and stationarity-enforcing reparameterizations (e.g., exponential transforms for scale parameters and logistic transforms to constrain persistence parameters to $(0, 1)$). One-step-ahead variance forecasts in the test set are then computed using lagged realized volatility only, i.e., using x_{t-1} but not x_t, so that the forecast is genuinely out-of-sample.

For the returns-only SV model, the training sample is used to draw from the joint posterior of parameters and latent states via MCMC. To produce out-of-sample volatility forecasts, the code adopts a two-stage procedure: posterior medians of the SV parameters are fixed at their training-sample estimates, and a bootstrap particle filter (Sequential Importance Resampling) is used to sequentially infer the latent log-volatility h_t over the test period.

The filter propagates a swarm of $N = 5000$ particles using the transition density of the latent state. At each time step t, importance weights are calculated based on the likelihood of the observation conditional on the state particles. Systematic resampling is performed at every step to mitigate particle degeneracy. This approach makes explicit the distinction emphasized in the SV literature between parameter estimation and state filtering in latent volatility models [6, 2].

For the realized-measure SV model, the training sample is used to estimate parameters and latent states in the joint returns–measurement state-space system via Bayesian MCMC, consistent with the realized-volatility SV approach in Takahashi et al. [12] and the measurement-error emphasis of Shirota and Omori [11]. Out-of-sample state inference in the test period is again conducted via particle filtering, but with weights proportional to the *joint* likelihood contribution of the return r_t and the realized measurement $\log x_t$. This yields a filtered latent volatility sequence that reflects both return information and realized volatility information, while still preserving an autonomous volatility innovation process.

3.4 Forecasting Protocol

All models are evaluated on one-step-ahead forecasts over the held-out test period. The information set at forecast origin $t - 1$ includes all past returns and, for realized-measure models, the realized variance up to time $t - 1$. Forecasts are produced recursively through the test set without using future information.

The primary volatility forecast output is $\hat{h}_t$, interpreted as the model-implied conditional variance of returns at time t. For reporting and visualization, we also consider model-implied volatility $\sqrt{\hat{h}_t}$ and compare it against $\sqrt{x_t}$, while keeping evaluation and losses on the variance scale to align with the model objects.

3.5 Evaluation Metrics

Following standard practice in volatility forecast comparisons [8], forecasting performance is assessed using realized volatility as a proxy for latent variance. Because x_t is a noisy proxy, conclusions may depend on the chosen loss function [8]; we therefore report multiple complementary criteria.

Let x_t denote the realized variance proxy and $\hat{h}_t$ the one-step-ahead variance forecast. We compute:

1. The QLIKE loss,
$$\ell_t^{\mathrm{QLIKE}}(\hat{h}_t) = \log(\hat{h}_t) + \frac{x_t}{\hat{h}_t}, \tag{3.1}$$
   which is widely used in volatility evaluation and is relatively robust to measurement

error in volatility proxies [8]. In practice, QLIKE penalizes forecasts that severely underpredict volatility due to the ratio term $x_t/\hat{h}_t$, which is relevant when comparing models that differ in responsiveness to volatility bursts.

2. Mean squared error (MSE) and mean absolute error (MAE),

$$\mathrm{MSE} = \frac{1}{H}\sum_{t=1}^{H}(x_t - \hat{h}_t)^2, \qquad \mathrm{MAE} = \frac{1}{H}\sum_{t=1}^{H}|x_t - \hat{h}_t|, \tag{3.2}$$

which measure average deviation of the variance forecast from the realized proxy on the original scale.

In addition to volatility forecast losses, we report the average Gaussian log predictive score for returns,

$$\mathrm{LogScore} = \frac{1}{H}\sum_{t=1}^{H}\left[-\frac{1}{2}\left(\log(2\pi) + \log(\hat{h}_t) + \frac{(r_t - \hat{\mu}_t)^2}{\hat{h}_t}\right)\right], \tag{3.3}$$

where $\hat{\mu}_t$ is the model-implied conditional mean (constant in the implementations used here). This scoring rule links volatility forecasts to return density forecasting, allowing us to check whether improvements in variance tracking translate into improved predictive likelihood for returns.

All reported metrics are computed over the same test window for all models using the same realized variance series, ensuring that comparisons reflect differences in model structure and information usage rather than differences in data handling. The complete implementation details, including parameter settings, filtering choices, and all metric computations, are documented in the code listing in Appendix A.

4 Numerical Results

This section reports the out-of-sample forecasting results for the four volatility specifications introduced in Section 3. Performance is evaluated over the test set using one-step-ahead forecasts, with realized volatility serving as a proxy for latent variance. We first summarize parameter estimates, then compare RV forecast accuracy under multiple loss functions, and finally interpret time-series diagnostics from Figures 1–2.

4.1 Estimated Parameters

Table 1 reports fitted parameter values for each model. For the returns-only GARCH(1,1), the estimated persistence is high ($\alpha_1 + \beta_1 \approx 0.992$), consistent with the well-documented slow decay of volatility shocks in equity index returns. The returns-only SV model similarly implies strong persistence, with ϕ close to unity, indicating that log-volatility evolves slowly over time even in the absence of realized measures.

For the realized-measure SV model, persistence is estimated to be extremely high ($\phi \approx 0.9996$). While this value is close to a unit root, it is a common finding in realized volatility modeling when the measurement equation successfully separates transient noise from the persistent component of volatility. In the returns-only SV model, the latent state must account for all return variation, often resulting in lower persistence estimates to capture short-term noise. In the realized-measure framework, the measurement error term σ_u absorbs the high-frequency transitory shocks (microstructure noise), leaving the latent h_t process to capture the highly persistent underlying trend of integrated variance. This separation of scales is a key theoretical advantage of the SV framework.

The Realized GARCH estimates display extreme values for (ω, β) under the chosen parameterization. Because these parameters are constrained through nonlinear transformations, their raw magnitudes are not directly interpretable in isolation. Instead, their implications should be assessed through the resulting volatility path and forecasting performance, which are examined below.

Table 1: Estimated model parameters on training set rounded up to necessary precision.

(a) GARCH(1,1)

μ	ω	α_1	β_1
4.23×10^{-4}	1.25×10^{-6}	7.95×10^{-2}	9.13×10^{-1}

(b) SV (stochvol)

μ_{return}	μ_h	ϕ	σ_η
-9.69×10^{-6}	-9.15	0.991	0.138

(c) Realized GARCH

μ	ω	β	γ	ξ	δ	σ_u
-1.10×10^{-4}	0	0	1.59	-2.18	0.810	0.613

(d) Realized SV (Stan)

μ	μ_h	ϕ	σ_η	α	σ_u
6.53×10^{-4}	-0.322	0.9996	0.216	-0.260	0.428

Table 2: Out-of-sample realized volatility forecast performance.

Model	QLIKE_{RV}	MSE_{RV}	MAE_{RV}
GARCH(1,1)	-8.55228	1.66044×10^{-8}	7.02934×10^{-5}
SV (filtered)	-8.60760	1.26463×10^{-8}	5.89348×10^{-5}
Realized GARCH	-8.77631	1.15478×10^{-8}	5.53729×10^{-5}
Realized SV (filtered)	-8.76418	5.44031×10^{-9}	3.66354×10^{-5}

4.2 Realized Volatility Forecast Accuracy

Table 2 compares out-of-sample volatility forecast accuracy using realized volatility as the evaluation proxy. In the returns-only setting, the SV model outperforms GARCH across QLIKE, MSE, and MAE, though the differences are relatively small. This confirms that latent volatility dynamics offer modest gains when inference relies solely on daily returns.

Incorporating realized volatility leads to large improvements for both realized-measure models. Relative to returns-only GARCH, the realized-measure SV model reduces RV forecast MSE by roughly two-thirds and MAE by nearly one-half, indicating a substantial increase in point forecast accuracy. Realized GARCH also delivers large gains relative to returns-only models.

However, the ranking of the two realized-measure models depends on the loss function. Realized GARCH attains the lowest QLIKE, while the realized-measure SV model achieves the lowest MSE and MAE. This divergence is consistent with the asymmetric nature of QLIKE, which penalizes underestimation of volatility more strongly than overestimation.

Figure 1 helps explain this result. The Realized GARCH implied volatility tracks realized volatility extremely closely and reacts aggressively to sharp spikes, often slightly overshooting realized volatility during turbulent periods. This behavior reduces the frequency and magnitude of under-predicted volatility, which is rewarded by QLIKE. In contrast, the realized-measure SV volatility path remains more centered around realized volatility and is less prone to extreme overshooting, which lowers average squared and absolute errors and explains its superior MSE and MAE performance.

The returns-only GARCH and SV models produce noticeably smoother volatility paths that do not attempt to match high-frequency spikes in realized volatility. While this smoothing limits responsiveness, it also avoids excessive noise, illustrating the fundamental trade-off between responsiveness and stability across model classes.

Table 3: Out-of-sample return density and mean forecast performance.

Model	LogScore$_r$	MSE$_\mu$	MAE$_\mu$
GARCH(1,1)	3.26176	1.16632×10^{-4}	7.46541×10^{-3}
SV (filtered)	3.35732	1.16808×10^{-4}	7.49263×10^{-3}
Realized GARCH	3.42243	1.16901×10^{-4}	7.50373×10^{-3}
Realized SV (filtered)	3.42340	1.16692×10^{-4}	7.46243×10^{-3}

4.3 Return Density and Mean Forecast Performance

Table 3 reports Gaussian log predictive scores for returns along with mean forecast errors. Differences in MSE$_\mu$ and MAE$_\mu$ are extremely small across all models, reflecting the fact that the conditional mean is modeled very simply and that performance differences arise primarily through variance forecasting.

The log predictive scores show a clearer ordering. Both realized-measure models outperform their returns-only counterparts, with the realized-measure SV model achieving the highest log score, albeit by a small margin. These gains are consistent with improved calibration of the conditional variance rather than changes in the conditional mean.

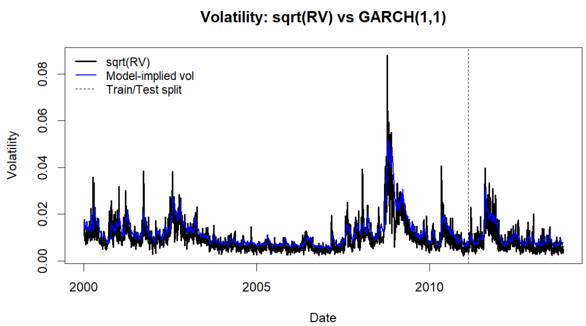
4.4 Time-Series Diagnostics from Volatility and Predictive Bands

Figure 1 provides a direct visual comparison of how each model tracks the timing and magnitude of volatility. The Realized GARCH specification produces the most reactive volatility path, closely following nearly all spikes in realized volatility and exhibiting pronounced jaggedness. The realized-measure SV model is also more responsive than returns-only models but remains smoother and more centered. This visual evidence aligns closely with the QLIKE–MSE trade-off observed in Table 2.

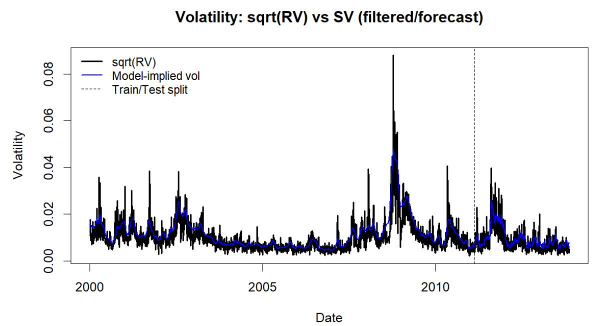
Figure 2 plots returns together with model-implied $\pm 2\sigma$ predictive bands. These bands represent conditional dispersion rather than explicit mean forecasts. For GARCH, SV, and realized-measure SV, the bands expand and contract smoothly over time and generally track volatility regimes without excessive short-term variation. By contrast, Realized GARCH produces extremely jagged and rapidly fluctuating bands, creating a dense appearance around the returns. This behavior mirrors the volatility dynamics in Figure 1 and reflects the model's strong sensitivity to realized volatility inputs.

Taken together, the results suggest that realized measures materially improve volatility forecasting in both deterministic and latent-state frameworks. The realized-measure SV model delivers more stable and centered volatility estimates with superior average error performance, while Realized GARCH provides more conservative coverage during volatility spikes, which is favored by QLIKE and reflected in its highly reactive volatility.

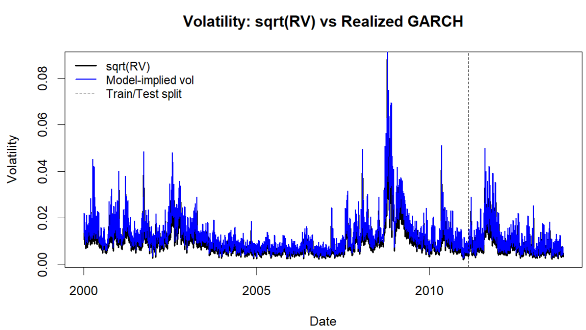
Figure 1: Volatility comparison: realized volatility vs model-implied volatility.



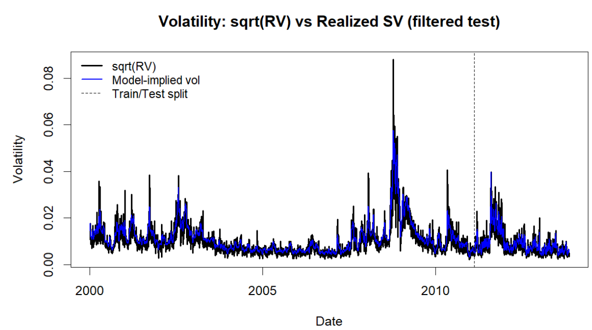
(a) GARCH(1,1)



(b) SV

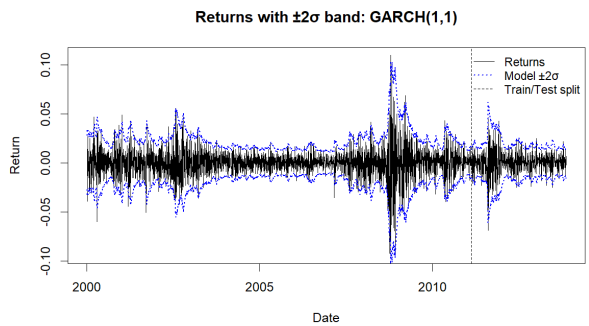


(c) Realized GARCH

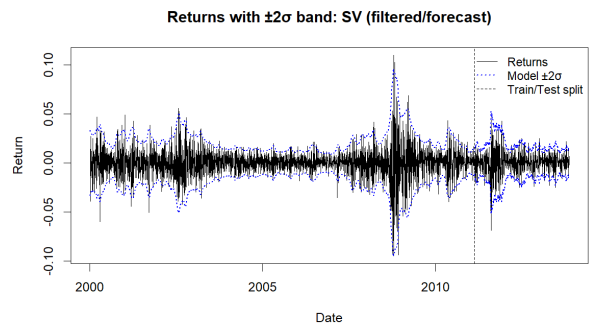


(d) Realized SV

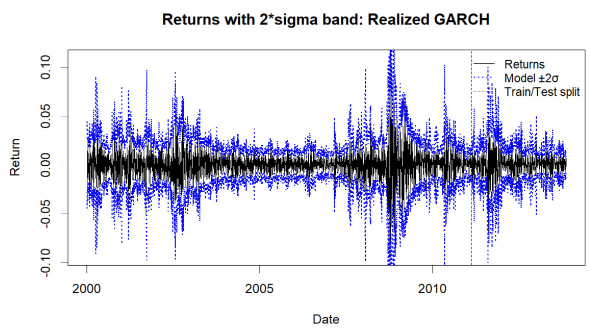
Figure 2: Returns with model-implied $\pm 2\sigma$ predictive bands.



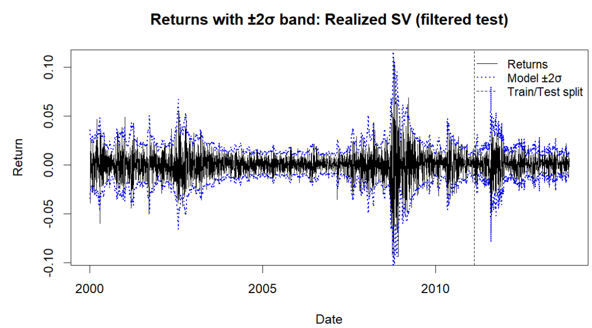
(a) GARCH(1,1)



(b) SV

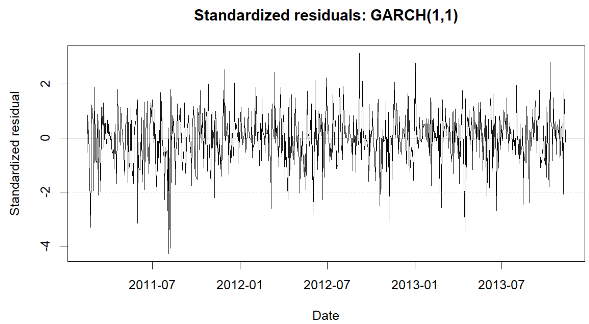


(c) Realized GARCH

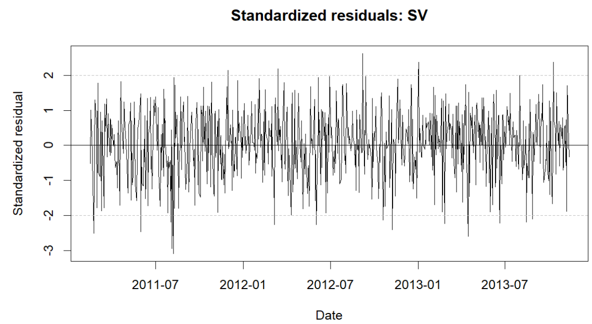


(d) Realized SV

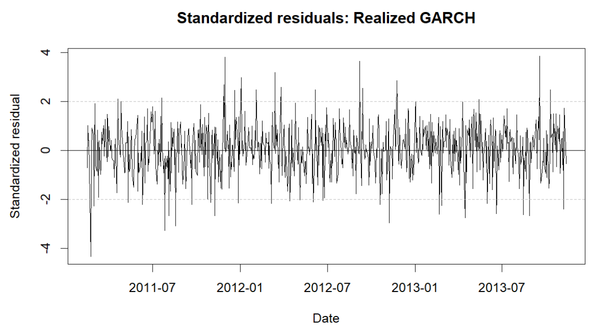
Figure 3: Standardized return residuals, $\hat{z}_t = (r_t - \hat{\mu}_t)/\sqrt{\hat{h}_t}$, over the test sample.



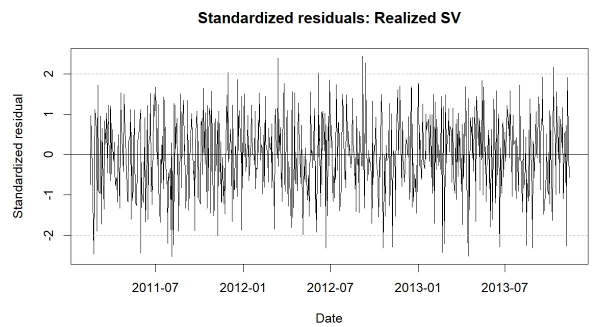
(a) GARCH(1,1)



(b) SV

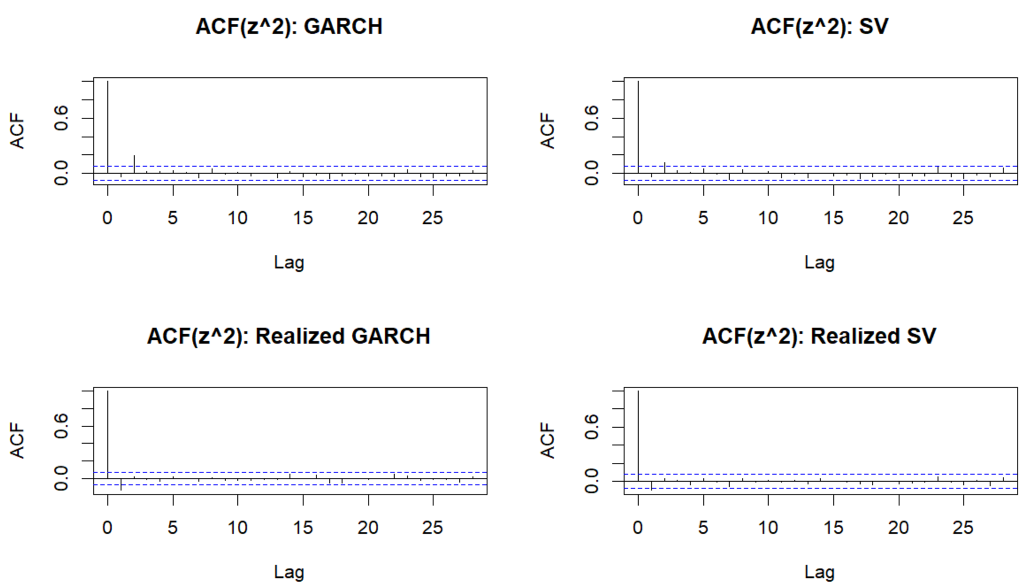


(c) Realized GARCH



(d) Realized SV

Figure 4: Autocorrelation functions of squared standardized residuals, $\hat{z}_t^2$, for each model.



5 Discussion and Theoretical Analysis

This section provides a theoretical interpretation of the numerical results reported in Section 4. Rather than introducing new models, the discussion focuses on explaining why the observed differences in volatility paths, forecast accuracy, loss-function rankings, and residual diagnostics arise as a consequence of model structure and information availability. The central theme is that incorporating realized volatility fundamentally changes the volatility inference problem, but that deterministic and latent-state frameworks respond to this information in systematically different ways.

5.1 Latent Variance, Proxies, and Forecast Error Decomposition

All four models considered in this paper aim to forecast the same latent object: the conditional variance of returns,

$$\sigma^2_{t+1} = \mathrm{Var}(r_{t+1} \mid \mathcal{F}_t),$$

where $\mathcal{F}_t$ denotes the information set available at time t. In practice, this quantity is unobserved, and model forecasts $\hat{h}_{t+1}$ must be evaluated using a proxy. Following much of the volatility forecasting literature, realized volatility is used as a measurement-based proxy for latent variance,

$$x_{t+1} = RV_{t+1} = \sigma^2_{t+1} + \varepsilon^{(RV)}_{t+1},$$

where $\varepsilon^{(RV)}_{t+1}$ captures measurement error arising from microstructure noise, finite sampling, and jump components [1, 3].

This representation implies a natural decomposition of the forecast error:

$$x_{t+1} - \hat{h}_{t+1} = (\sigma^2_{t+1} - \hat{h}_{t+1}) + \varepsilon^{(RV)}_{t+1}.$$

Even if a model were to forecast the latent variance perfectly, evaluation against realized volatility would still exhibit error due to the measurement component. This decomposition helps explain why differences in loss-function rankings can persist even when residual diagnostics suggest that conditional heteroskedasticity has largely been removed.

5.2 Information Sets and the Identifiability of Volatility

In returns-only models, volatility must be inferred indirectly from daily returns. Large return realizations may be driven either by genuinely high conditional variance or by unusually large innovations, and this confounding limits identifiability of the latent volatility process [8]. As a result, both GARCH and stochastic volatility models estimated on returns alone rely on long-run persistence and smoothing to recover volatility dynamics.

This limitation is clearly visible in the volatility paths in Figure 1. Returns-only GARCH and SV models capture broad volatility regimes but do not track short-lived spikes in realized volatility. The modest improvement of returns-only SV over GARCH in Table 2 reflects the fact that latent volatility innovations add flexibility, but cannot fully overcome the weak information content of daily returns.

In contrast, realized-measure models explicitly condition on realized volatility, dramatically expanding the information set. By incorporating x_t directly into the volatility updating mechanism or measurement equation, these models reduce uncertainty about the latent variance and respond more rapidly to changes in market conditions. The large improvements in RV forecast accuracy for both realized-measure models are therefore best interpreted as informational gains rather than purely parametric improvements.

5.3 Deterministic Updating versus Stochastic Volatility Dynamics

Although both realized-measure models exploit the same realized volatility signal, they differ fundamentally in how volatility evolves conditional on information. In Realized GARCH, the conditional variance is updated deterministically:

$$h_{t+1} = g(h_t, x_t),$$

implying

$$\mathrm{Var}(h_{t+1} \mid \mathcal{F}_t) = 0.$$

Once parameters are fixed, all uncertainty about future volatility is resolved by observed past data.

By contrast, realized-measure SV models retain autonomous volatility innovations:

$$h_{t+1} = \mu_h + \phi(h_t - \mu_h) + \eta_{t+1}, \qquad \eta_{t+1} \sim \mathcal{N}(0, \sigma_\eta^2),$$

so that

$$\mathrm{Var}(h_{t+1} \mid \mathcal{F}_t) = \sigma_\eta^2 > 0.$$

Even conditional on realized volatility, uncertainty about future volatility remains. This structural difference implies that Realized GARCH can react arbitrarily strongly to realized volatility movements, while realized-measure SV necessarily smooths volatility updates through latent-state uncertainty.

This distinction explains the volatility plots in Figure 1. Realized GARCH closely tracks nearly all spikes in realized volatility and often overshoots them, producing a highly jagged volatility path. The realized-measure SV model is more responsive than returns-only models, but remains smoother and more centered around realized volatility.

5.4 Parameter Degeneracy and Information Dominance in Realized GARCH

An important empirical feature of the Realized GARCH estimates is that the traditional GARCH parameters governing the conditional variance recursion become economically negligible once realized volatility is incorporated. In particular, the estimated coefficients on lagged squared returns and lagged conditional variance shrink toward zero, while the realized volatility measurement equation accounts for the bulk of volatility dynamics. To see why this occurs, consider the Realized GARCH variance recursion

$$h_t = \omega + \beta h_{t-1} + \gamma x_{t-1}.$$

When the realized volatility measure x_{t-1} is highly informative about latent variance, maximum-likelihood estimation assigns most of the explanatory weight to the realized-measure channel. In this case, the persistence and intercept terms (ω, β) play only a minor role, and the conditional variance is driven primarily by x_{t-1}. This behavior reflects information dominance rather than numerical instability or misspecification: once volatility is effectively observed up to measurement error, the traditional GARCH recursion becomes largely redundant. Similar parameter shrinkage is documented in the Realized GARCH literature when realized measures are sufficiently informative [9].

This behavior reflects the informational dominance of realized volatility measures rather than a failure of the GARCH framework. When volatility is inferred solely from returns, the GARCH recursion serves as a reduced-form mechanism for extracting persistence from noisy squared returns. However, once a high-frequency-based proxy for integrated variance is introduced, the latent volatility process becomes effectively observed, up to measurement error. In this setting, there is little remaining role for deterministic variance updating based on past returns.

Similar findings are documented in the Realized GARCH literature, where realized measures often subsume the explanatory power of traditional GARCH components when they are sufficiently informative [9]. From an information-set perspective, the realized volatility augments the model with a near-direct observation of the latent variance, rendering the return-driven updating mechanism largely redundant.

5.5 Loss-Function Dependence and the QLIKE–MSE Trade-off

The divergence between QLIKE and MSE/MAE rankings in Table 2 follows directly from the structure of the loss functions. QLIKE evaluates forecasts according to

$$\ell_t(\hat{h}_t) = \log(\hat{h}_t) + \frac{x_t}{\hat{h}_t},$$

which penalizes underestimation of volatility more heavily than overestimation. In particular, when realized volatility is high, forecasts that err on the side of larger $\hat{h}_t$ incur smaller losses than forecasts that undershoot the variance.

Realized GARCH's tendency to react aggressively to realized volatility spikes and to slightly overestimate variance during turbulent periods reduces the expected contribution of the $\frac{x_t}{\hat{h}_t}$ term, yielding a lower QLIKE. By contrast, MSE and MAE penalize deviations symmetrically:

$$\mathbb{E}[(x_t - \hat{h}_t)^2], \qquad \mathbb{E}[|x_t - \hat{h}_t|],$$

so that overshooting volatility is costly. The more centered and smoother volatility path produced by realized-measure SV therefore leads to lower MSE and MAE, even though it does not achieve the minimum QLIKE.

These results illustrate that apparent model dominance depends critically on the evaluation criterion, a point emphasized by Hansen and Lunde [8]. The choice of loss function should therefore reflect the decision problem faced by the practitioner.

5.6 Standardized Residual Diagnostics and Conditional Calibration

Standardized residuals,

$$\hat{z}_t = \frac{r_t - \hat{\mu}_t}{\sqrt{\hat{h}_t}},$$

provide a direct diagnostic of variance calibration. If the conditional variance model is correctly specified, $\hat{z}_t$ should be approximately i.i.d. with unit variance.

The standardized residual plots in Figure 3 show that returns-only models exhibit more frequent large residuals, indicating incomplete normalization of return variance. Realized GARCH produces notably compressed residuals with fewer extreme excursions, consistent with its aggressive variance scaling. However, this compression reflects deterministic overreaction to realized volatility and does not imply superior point accuracy under symmetric losses.

The realized-measure SV model produces residuals that are well centered and moderately dispersed, suggesting a better balance between responsiveness and stability. Importantly, the autocorrelation functions of squared standardized residuals in Figure 4 indicate that all models largely eliminate serial dependence in $\hat{z}_t^2$. Differences across models therefore arise from variance level calibration rather than from remaining conditional heteroskedasticity.

5.7 Filtering, Measurement Error, and Volatility Tracking

The contrasting behavior of the realized-measure models can be further understood through the lens of filtering in the presence of noisy observations. Realized volatility provides a highly informative but imperfect signal about latent variance, and models differ primarily in how strongly they map this signal into volatility updates.

In Realized GARCH, realized volatility enters directly into the variance recursion. Abstracting from functional form, the update can be written as

$$h_{t+1} = g(h_t, x_t),$$

so that realized volatility is treated as a state-determining input rather than a noisy measurement. As a result, high-frequency variation in x_t is transmitted almost one-for-one into h_{t+1}. This produces a volatility path that closely tracks realized volatility but also inherits its high-frequency noise and jump-induced variation.

By contrast, the realized-measure SV model embeds realized volatility in a measurement equation,

$$\log x_t = \alpha + h_t + u_t, \qquad u_t \sim \mathcal{N}(0, \sigma_u^2),$$

while the latent volatility evolves independently according to a stochastic state equation. Filtering combines information from returns, realized volatility, and the state dynamics to produce an estimate of h_t that minimizes expected squared error conditional on the assumed model. In effect, realized volatility is downweighted according to its estimated measurement error variance σ_u^2, preventing individual spikes in x_t from fully propagating into the latent volatility path.

This distinction explains the volatility comparisons in Figure 1. Realized GARCH behaves as a high-gain filter that aggressively tracks realized volatility at the cost of introducing excess variability. The realized-measure SV model behaves as a lower-gain filter, producing a smoother volatility path that remains responsive to sustained changes in realized volatility but attenuates transient fluctuations.

Importantly, this difference is not a matter of misspecification but reflects a deliberate modeling choice. Treating realized volatility as an observed regressor prioritizes tracking accuracy, while treating it as a noisy measurement prioritizes stability of the latent state. The relative performance of the two approaches therefore depends on whether the evaluation criterion rewards responsiveness or penalizes variability.

6 Conclusion

This paper examined the empirical and theoretical differences between GARCH and stochastic volatility models when volatility is inferred from daily returns alone and when realized volatility measures are incorporated as additional information. By maintaining a common dataset, forecasting design, and evaluation framework, the analysis isolates how model structure and information usage drive performance rather than differences in implementation.

6.1 Main Findings and Contributions

The primary contribution of this paper is to demonstrate that the apparent empirical similarity between GARCH and stochastic volatility models in returns-only settings is largely a consequence of limited volatility observability. When volatility must be inferred solely from daily returns, both frameworks rely on strong persistence and smoothing, leading to broadly comparable forecasting performance.

Once realized volatility is incorporated, this equivalence breaks down. In particular, the Realized GARCH results indicate that when realized volatility is highly informative, the model effectively operates as a measurement-driven variance updater rather than as a returns-based GARCH process.

Both realized-measure models deliver substantial improvements in realized volatility forecast accuracy, confirming that high-frequency-based measures provide meaningful information about latent variance. However, the relative ranking of realized GARCH and realized-measure stochastic volatility depends on the evaluation criterion. Realized GARCH achieves the lowest QLIKE, reflecting its aggressive reaction to volatility spikes, while realized-measure stochastic volatility achieves lower MSE and MAE by producing smoother and more stable variance forecasts.

These results highlight that differences across model classes become empirically relevant only once volatility is partially observed, and that no single specification dominates across all loss functions.

6.2 Limitations of the Empirical Design

Several limitations of the present empirical design should be acknowledged. First, the analysis focuses exclusively on one-step-ahead forecasts. While this horizon is standard in the volatility forecasting literature, longer-horizon forecasts may amplify differences in persistence and filtering behavior across models.

Second, the study considers a single equity index. Although the S&P 500 provides a natural benchmark, asset classes with different trading intensity, jump behavior, or leverage effects may yield different relative performance patterns.

Third, Gaussian innovations were adopted throughout to maintain comparability across models. Allowing for heavy-tailed return distributions or asymmetric innovations could further affect density-based evaluation and residual diagnostics.

6.3 Limitations of the Modeling Frameworks and Literature

The results also reflect limitations inherent in the modeling frameworks drawn from the literature. Realized GARCH models treat realized volatility as a state-determining input, implicitly assuming that realized measures are sufficiently accurate to drive variance dynamics directly. This can lead to amplification of measurement noise, particularly during periods of market stress.

Conversely, realized-measure stochastic volatility models rely on latent-state filtering that smooths high-frequency variation. While this improves stability, it may delay responsiveness to abrupt volatility shifts. These trade-offs are structural and reflect modeling choices emphasized in the respective literatures rather than estimation shortcomings.

More broadly, evaluation based on realized volatility proxies remains imperfect. Even robust loss functions such as QLIKE cannot fully disentangle forecast error in latent variance from measurement error in realized volatility, a limitation noted throughout the volatility forecasting literature.

6.4 Directions for Future Research

Several extensions could build on the present analysis. First, alternative realized measures that are more robust to jumps and microstructure noise could be incorporated to assess the sensitivity of results to the choice of volatility proxy.

Second, extending the comparison to multi-step-ahead forecasts and multivariate settings would provide insight into how model structure affects volatility dynamics over longer horizons and across assets.

Finally, integrating richer return dynamics, such as heavy-tailed or asymmetric innovations, may further clarify how volatility modeling choices interact with density forecasting performance.

Overall, the findings emphasize that empirical comparisons of volatility models are highly sensitive to information availability and evaluation criteria. Once volatility is partially observed through realized measures, the distinction between deterministic variance updating and latent stochastic dynamics becomes empirically meaningful, and model choice should be guided by the specific objectives of the forecasting task.

References

[1] Torben G. Andersen, Tim Bollerslev, Francis X. Diebold, and Paul Labys. Modeling and forecasting realized volatility. *Econometrica*, 71(2):579–625, 2003.

[2] Manabu Asai, Michael McAleer, and Marcelo C. Medeiros. Asymmetry and leverage in stochastic volatility models. *Journal of Financial Econometrics*, 10(2):1–40, 2012.

[3] Ole E. Barndorff-Nielsen and Neil Shephard. Power and bipower variation with stochastic volatility and jumps. *Journal of Financial Econometrics*, 2(1):1–37, 2004.

[4] Tim Bollerslev. Generalized autoregressive conditional heteroskedasticity. *Journal of Econometrics*, 31(3):307–327, 1986.

[5] Fulvio Corsi. A simple approximate long-memory model of realized volatility. *Journal of Financial Econometrics*, 7(2):174–196, 2009.

[6] Eric Ghysels, Andrew Harvey, and Eric Renault. Stochastic volatility. *Handbook of Statistics*, 14:119–191, 1996.

[7] Eric Ghysels, Arthur Sinko, and Rossen Valkanov. Mixed frequency data sampling regression models: The R package `midasr`. *Journal of Statistical Software*, 72(4):1–33, 2016.

[8] Peter R. Hansen and Asger Lunde. A forecast comparison of volatility models: Does anything beat a garch(1,1)? *Journal of Applied Econometrics*, 20(7):873–889, 2005.

[9] Peter R. Hansen, Zhuo Huang, and Howard H. Shek. Realized garch: A joint model for returns and realized measures. *Journal of Applied Econometrics*, 27(6):877–906, 2012.

[10] Neil Shephard and Kevin Sheppard. Realising the future: Forecasting with high-frequency-based volatility (heavy) models. *Journal of Applied Econometrics*, 25(2): 197–231, 2010.

[11] Shinji Shirota and Yasuhiro Omori. Measurement error stochastic volatility models. *Computational Statistics & Data Analysis*, 76:151–167, 2014.

[12] Masayuki Takahashi, Yasuhiro Omori, and Toshiaki Watanabe. Estimating stochastic volatility models using daily returns and realized volatility. *Computational Statistics & Data Analysis*, 53(6):2404–2426, 2009.

[13] Stephen J. Taylor. *Modelling Financial Time Series*. Wiley, 1986.

A Supplementary Code

Listing 1: Full code: data, estimation, forecasts, and evaluation

```
packages <-
  c("midasr","quantmod","xts","rugarch","stochvol","rstan")
to_install <- packages[!(packages %in%
  installed.packages()[,"Package"])]
if(length(to_install)>0) install.packages(to_install)

library(midasr); library(quantmod); library(xts)
library(rugarch); library(stochvol); library(rstan)

rstan_options(auto_write = TRUE)
options(mc.cores = 10)
set.seed(1)

# ---------- PF helper functions
logsumexp <- function(a){
  m <- max(a)
  m + log(sum(exp(a - m)))
}

resample_systematic <- function(w){
  N <- length(w)
  u0 <- runif(1) / N
  u <- u0 + (0:(N-1)) / N
  cdf <- cumsum(w)
  idx <- integer(N)
  j <- 1
  for(i in 1:N){
    while(u[i] > cdf[j]) j <- j + 1
    idx[i] <- j
  }
  idx
}

# ---------- Data
data("rvsp500", package="midasr")
rv_dates <- as.Date(as.character(rvsp500$DateID), format="%Y%m%d")
rv_xts <- xts(as.numeric(rvsp500$SPX2.rv), order.by=rv_dates)
colnames(rv_xts) <- "RV"
```

```r
getSymbols("^GSPC", src="yahoo", from="2000-01-01",
    to="2013-12-31", auto.assign=TRUE)
px <- Ad(GSPC)
r_xts <- diff(log(px))
colnames(r_xts) <- "r"
index(r_xts) <- as.Date(index(r_xts))

dat <- merge(r_xts, rv_xts, join="inner")
dat <- dat[is.finite(dat$r) & is.finite(dat$RV) & !is.na(dat$r) &
    !is.na(dat$RV) & dat$RV > 0, ]
dat$logRV <- log(dat$RV)

stopifnot(
  NROW(dat) > 0,
  all(is.finite(coredata(dat$r))),
  all(is.finite(coredata(dat$RV))),
  all(coredata(dat$RV) > 0)
)

dates <- index(dat)
n <- NROW(dat)
test_n <- max(1, floor(0.2*n))
train_n <- n - test_n

train <- dat[1:train_n]
test  <- dat[(train_n+1):n]
split_date <- dates[train_n]

r_all  <- as.numeric(dat$r)
x_all  <- as.numeric(dat$RV)

r_train  <- as.numeric(train$r)
r_test   <- as.numeric(test$r)
x_train  <- as.numeric(train$RV)
x_test   <- as.numeric(test$RV)
lx_train <- as.numeric(train$logRV)
lx_test  <- as.numeric(test$logRV)

H <- length(r_test)

```

```r
# ---------- Metrics
qlike <- function(y, hhat) mean(log(hhat) + y/hhat, na.rm=TRUE)
mse   <- function(y, yhat) mean((y - yhat)^2, na.rm=TRUE)
mae   <- function(y, yhat) mean(abs(y - yhat), na.rm=TRUE)

gauss_logscore <- function(r, mu, h){
  mean(-0.5*(log(2*pi) + log(h) + (r - mu)^2/h), na.rm=TRUE)
}

# ---------- GARCH(1,1)
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"
)
garch_fit <- ugarchfit(spec=garch_spec, data=r_train,
   solver="hybrid")

cf_g <- coef(garch_fit)
omega_g <- unname(cf_g["omega"])
alpha_g <- unname(cf_g["alpha1"])
beta_g  <- unname(cf_g["beta1"])
mu_g    <- if ("mu" %in% names(cf_g)) unname(cf_g["mu"]) else
   mean(r_train)

h_g_train <- as.numeric(sigma(garch_fit))^2
h_last_g  <- tail(h_g_train, 1)

h_g_test <- rep(NA_real_, H)
h_prev <- as.numeric(h_last_g)
r_prev <- tail(r_train, 1)
for(i in 1:H){
  eps_prev <- (if(i==1) r_prev else r_test[i-1]) - mu_g
  h_prev <- omega_g + alpha_g*(eps_prev^2) + beta_g*h_prev
  h_g_test[i] <- h_prev
}

h_g_all <- c(h_g_train, h_g_test)
mu_g_all <- rep(mu_g, n)

# ---------- SV fit (stochvol) + in-sample variance
```

```
r_train_mu <- mean(r_train)
r_train_dm <- r_train - r_train_mu

sv_fit <- svsample(r_train_dm, draws=4000, burnin=1000, quiet=TRUE)

pars_sv <- as.matrix(sv_fit$para)
mu_sv_med  <- median(pars_sv[,"mu"])
phi_sv_med <- median(pars_sv[,"phi"])
sig_sv_med <- median(pars_sv[,"sigma"])

latent_sv <- as.matrix(sv_fit$latent)  # draws x T
h_sv_train <- apply(exp(latent_sv), 2, median)

# ---------- SV particle filter on test (fixed params; no refit)
mu_h_pf    <- mu_sv_med
phi_pf     <- phi_sv_med
sig_eta_pf <- sig_sv_med

Np_sv <- 5000

hT_sv <- latent_sv[, ncol(latent_sv)]
h_particles <- sample(hT_sv, size=Np_sv, replace=TRUE)

h_filt_test_sv <- numeric(H)

for(t in 1:H){
  h_particles <- mu_h_pf + phi_pf*(h_particles - mu_h_pf) +
     rnorm(Np_sv, 0, sig_eta_pf)

  rt_dm <- r_test[t] - r_train_mu
  loglik_r <- dnorm(rt_dm, mean=0, sd=exp(h_particles/2), log=TRUE)

  logw <- loglik_r
  logw <- logw - logsumexp(logw)
  w <- exp(logw)

  h_filt_test_sv[t] <- sum(w * h_particles)

  idx <- resample_systematic(w)
  h_particles <- h_particles[idx]
}
```

```
h_sv_test <- exp(h_filt_test_sv)
h_sv_all <- c(h_sv_train, h_sv_test)
mu_sv_all <- rep(r_train_mu, n)

# ---------- Realized GARCH (custom likelihood)
rgarch_nll <- function(par, r, x){
  omega <- exp(par[1])
  beta  <- 1/(1+exp(-par[2]))
  gamma <- exp(par[3])
  xi    <- par[4]
  delta <- par[5]
  su    <- exp(par[6])
  mu    <- par[7]

  n <- length(r)
  h <- rep(var(r), n)
  ll <- 0

  for(t in 2:n){
    h[t] <- omega + beta*h[t-1] + gamma*x[t-1]
    if(h[t] <= 0 || !is.finite(h[t])) return(1e12)

    eps <- r[t] - mu
    ll_r <- -0.5*(log(2*pi) + log(h[t]) + (eps^2)/h[t])

    m <- xi + delta*log(h[t])
    lx <- log(x[t])
    if(!is.finite(m) || !is.finite(lx)) return(1e12)

    ll_x <- -0.5*(log(2*pi) + 2*log(su) + ((lx - m)^2)/(su^2))
    if(!is.finite(ll_x)) return(1e12)

    ll <- ll + ll_r + ll_x
    if(!is.finite(ll)) return(1e12)
  }
  -ll
}

mu_hat <- mean(r_train)
rg_init <- c(log(0.01), qlogis(0.9), log(0.1), 0, 1, log(0.2),
```

```
    mu_hat)
rg_fit <- optim(rg_init, rgarch_nll, r=r_train, x=x_train,
    method="BFGS",
                  control=list(maxit=800))
rg_par <- rg_fit$par

omega_rg <- exp(rg_par[1])
beta_rg  <- 1/(1+exp(-rg_par[2]))
gamma_rg <- exp(rg_par[3])
xi_rg    <- rg_par[4]
delta_rg <- rg_par[5]
su_rg    <- exp(rg_par[6])
mu_rg    <- rg_par[7]

h_rg_train <- rep(var(r_train), length(r_train))
for(t in 2:length(r_train)){
  h_rg_train[t] <- omega_rg + beta_rg*h_rg_train[t-1] +
      gamma_rg*x_train[t-1]
}

# one-step-ahead: lagged RV only
h_rg_test <- rep(NA_real_, H)
h_prev <- tail(h_rg_train, 1)
x_prev <- tail(x_train, 1)
for(i in 1:H){
  x_lag <- if(i==1) x_prev else x_test[i-1]
  h_prev <- omega_rg + beta_rg*h_prev + gamma_rg*x_lag
  h_rg_test[i] <- h_prev
}

h_rg_all <- c(h_rg_train, h_rg_test)
mu_rg_all <- rep(mu_rg, n)

# ---------- Realized SV in Stan (fit on train)
stan_code <- "
data {
  int<lower=1> T;
  vector[T] r;
  vector[T] logx;
}
parameters {
```

```
  real mu;
  real mu_h;
  real<lower=-1,upper=1> phi;
  real<lower=0> sigma_eta;
  real alpha;
  real<lower=0> sigma_u;
  vector[T] h;
}
model {
  mu ~ normal(0, 1);
  mu_h ~ normal(0, 1);
  phi ~ normal(0, 0.5);
  sigma_eta ~ cauchy(0, 0.5);
  alpha ~ normal(0, 1);
  sigma_u ~ cauchy(0, 0.5);

  h[1] ~ normal(mu_h, sigma_eta / sqrt(1 - phi*phi));
  for(t in 2:T)
    h[t] ~ normal(mu_h + phi*(h[t-1]-mu_h), sigma_eta);

  r ~ normal(mu, exp(h/2));
  logx ~ normal(alpha + h, sigma_u);
}
"

stan_dat <- list(T=length(r_train), r=r_train, logx=lx_train)
sm <- stan_model(model_code=stan_code)

rsv_fit <- sampling(
  object=sm,
  data=stan_dat,
  chains=4,
  iter=2500,
  warmup=1500,
  seed=1,
  refresh=100,
  control=list(adapt_delta=0.99, max_treedepth=15)
)

post <- rstan::extract(rsv_fit,
   pars=c("mu","mu_h","phi","sigma_eta","alpha","sigma_u","h"))
```

```r

mu_rsv_med    <- median(post$mu)
mu_h_med      <- median(post$mu_h)
phi_med       <- median(post$phi)
sig_eta_med   <- median(post$sigma_eta)
alpha_med     <- median(post$alpha)
sig_u_med     <- median(post$sigma_u)

h_draws <- post$h
h_rsv_train <- apply(exp(h_draws), 2, median)
hT_draw2 <- h_draws[, ncol(h_draws)]

# ---------- Realized SV particle filter on test
Np_rsv <- 5000

mu_pf      <- mu_rsv_med
mu_h_pf2   <- mu_h_med
phi_pf2    <- phi_med
sig_eta_pf2<- sig_eta_med
alpha_pf   <- alpha_med
sig_u_pf2  <- sig_u_med

h_particles <- sample(hT_draw2, size=Np_rsv, replace=TRUE)
h_filt_test_rsv <- numeric(H)

for(t in 1:H){
  h_particles <- mu_h_pf2 + phi_pf2*(h_particles - mu_h_pf2) +
     rnorm(Np_rsv, 0, sig_eta_pf2)

  rt  <- r_test[t]
  lxt <- lx_test[t]

  loglik_r <- dnorm(rt, mean=mu_pf, sd=sqrt(exp(h_particles)),
     log=TRUE)
  loglik_x <- dnorm(lxt, mean=alpha_pf + h_particles, sd=sig_u_pf2,
     log=TRUE)

  logw <- loglik_r + loglik_x
  logw <- logw - logsumexp(logw)
  w <- exp(logw)

```

```
  h_filt_test_rsv[t] <- sum(w * h_particles)

  idx <- resample_systematic(w)
  h_particles <- h_particles[idx]
}

h_rsv_test <- exp(h_filt_test_rsv)
h_rsv_all  <- c(h_rsv_train, h_rsv_test)
mu_rsv_all <- rep(mu_rsv_med, n)

# ---------- Metric table (test)
rv_test <- x_test
models <- c("GARCH(1,1)", "SV", "RealizedGARCH", "RealizedSV")

rv_tbl <- data.frame(
  model = models,
  QLIKE_RV = c(
    qlike(rv_test, h_g_test),
    qlike(rv_test, h_sv_test),
    qlike(rv_test, h_rg_test),
    qlike(rv_test, h_rsv_test)
  ),
  MSE_RV = c(
    mse(rv_test, h_g_test),
    mse(rv_test, h_sv_test),
    mse(rv_test, h_rg_test),
    mse(rv_test, h_rsv_test)
  ),
  MAE_RV = c(
    mae(rv_test, h_g_test),
    mae(rv_test, h_sv_test),
    mae(rv_test, h_rg_test),
    mae(rv_test, h_rsv_test)
  )
)

ret_tbl <- data.frame(
  model = models,
  LogScore_ret = c(
    gauss_logscore(r_test, mu_g, h_g_test),
    gauss_logscore(r_test, r_train_mu, h_sv_test),
```

```r
    gauss_logscore(r_test, mu_rg, h_rg_test),
    gauss_logscore(r_test, mu_rsv_med, h_rsv_test)
  ),
  MSE_mean = c(
    mse(r_test, rep(mu_g, length(r_test))),
    mse(r_test, rep(r_train_mu, length(r_test))),
    mse(r_test, rep(mu_rg, length(r_test))),
    mse(r_test, rep(mu_rsv_med, length(r_test)))
  ),
  MAE_mean = c(
    mae(r_test, rep(mu_g, length(r_test))),
    mae(r_test, rep(r_train_mu, length(r_test))),
    mae(r_test, rep(mu_rg, length(r_test))),
    mae(r_test, rep(mu_rsv_med, length(r_test)))
  )
)

metric_tab <- merge(rv_tbl, ret_tbl, by="model", sort=FALSE)
print(metric_tab, row.names=FALSE, digits=6)

# ---------- Save per-model plots (Volatility and Returns)

save_vol_plot <- function(fname, title, vol_model, vol_real, dates,
   split_date){
  png(filename=fname, width=1200, height=700, res=150)
  plot(dates, vol_real, type="l", col="black", lwd=2,
       xlab="Date", ylab="Volatility",
       main=title)
  lines(dates, vol_model, col="blue", lwd=1.8)
  abline(v=split_date, lty=2)
  legend("topleft",
         legend=c("sqrt(RV)","Model-implied volatility","Train/Test
            split"),
         col=c("black","blue","black"),
         lty=c(1,1,2),
         lwd=c(2,1.8,1),
         bty="n",
         cex=0.9)
  dev.off()
}

```

```
save_ret_plot <- function(fname, title, r_all, mu_all, h_all,
    dates, split_date){
  png(filename=fname, width=1200, height=700, res=150)
  plot(dates, r_all, type="l", col="black",
       xlab="Date", ylab="Return",
       main=title)
  lines(dates, mu_all + 2*sqrt(h_all), col="blue", lty=3, lwd=1.4)
  lines(dates, mu_all - 2*sqrt(h_all), col="blue", lty=3, lwd=1.4)
  abline(v=split_date, lty=2)
  legend("topright",
         legend=c("Returns","Model pm 2*sigma","Train/Test split"),
         col=c("black","blue","black"),
         lty=c(1,3,2),
         lwd=c(1,1.4,1),
         bty="n",
         cex=0.9)
  dev.off()
}

# ---------- Volatility series
vol_real <- sqrt(x_all)

vol_g   <- sqrt(h_g_all)
vol_sv  <- sqrt(h_sv_all)
vol_rg  <- sqrt(h_rg_all)
vol_rsv <- sqrt(h_rsv_all)

# ---------- Volatility plots
save_vol_plot("vol_garch.png",
              "Volatility: sqrt(RV) vs GARCH(1,1)",
              vol_g, vol_real, dates, split_date)

save_vol_plot("vol_sv.png",
              "Volatility: sqrt(RV) vs SV (filtered)",
              vol_sv, vol_real, dates, split_date)

save_vol_plot("vol_rgarch.png",
              "Volatility: sqrt(RV) vs Realized GARCH",
              vol_rg, vol_real, dates, split_date)

save_vol_plot("vol_rsv.png",
```

```
              "Volatility: sqrt(RV) vs Realized SV (filtered)",
              vol_rsv, vol_real, dates, split_date)

# ---------- Return plots
save_ret_plot("ret_garch.png",
              "Returns with 2*sigma band: GARCH(1,1)",
              r_all, mu_g_all, h_g_all, dates, split_date)

save_ret_plot("ret_sv.png",
              "Returns with 2*sigma band: SV (filtered)",
              r_all, mu_sv_all, h_sv_all, dates, split_date)

save_ret_plot("ret_rgarch.png",
              "Returns with 2*sigma band: Realized GARCH",
              r_all, mu_rg_all, h_rg_all, dates, split_date)

save_ret_plot("ret_rsv.png",
              "Returns with 2*sigma band: Realized SV (filtered)",
              r_all, mu_rsv_all, h_rsv_all, dates, split_date)

cat("Saved plots:\n",
    "vol_garch.png, vol_sv.png, vol_rgarch.png, vol_rsv.png\n",
    "ret_garch.png, ret_sv.png, ret_rgarch.png, ret_rsv.png\n")

# ---------- Standardized residuals
z_garch <- (r_test - mu_g) / sqrt(h_g_test)
z_sv    <- (r_test - r_train_mu) / sqrt(h_sv_test)
z_rg    <- (r_test - mu_rg) / sqrt(h_rg_test)
z_rsv   <- (r_test - mu_rsv_med) / sqrt(h_rsv_test)

# ---------- Residual plots
save_resid_plot <- function(fname, title, z, dates){
  png(fname, width=1200, height=700, res=150)
  plot(dates, z, type="l", col="black",
       main=title, xlab="Date", ylab="Standardized residual")
  abline(h=c(-2,0,2), lty=c(2,1,2), col=c("gray","black","gray"))
  dev.off()
}

save_resid_plot("resid_garch.png", "Standardized residuals: GARCH(1,1)", z_garch, index(test))
```

```r
save_resid_plot("resid_sv.png",    "Standardized residuals: SV",
   z_sv, index(test))
save_resid_plot("resid_rgarch.png","Standardized residuals:
   Realized GARCH", z_rg, index(test))
save_resid_plot("resid_rsv.png",   "Standardized residuals:
   Realized SV", z_rsv, index(test))

# Residual ACF plot
png("resid_acf_sq.png", width=1200, height=700, res=150)
par(mfrow=c(2,2))
acf(z_garch^2, main="ACF(z^2): GARCH")
acf(z_sv^2,    main="ACF(z^2): SV")
acf(z_rg^2,    main="ACF(z^2): Realized GARCH")
acf(z_rsv^2,   main="ACF(z^2): Realized SV")
dev.off()
```